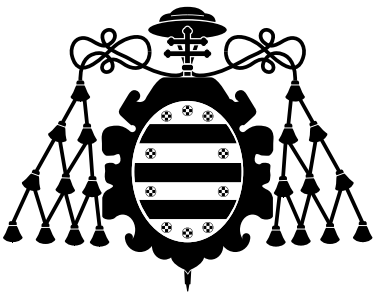


Universidad de Oviedo
*Universidá d'Uviéu*
*University of Oviedo*

Departamento de Informática

Tesis doctoral

# Estrategias heurísticas de planificación dinámica de trabajos con límite temporal para la optimización de recursos en entornos de Cloud Computing

Víctor Manuel Peláez Martínez

2019

Universidad de Oviedo
*Universidá d'Uviéu*
*University of Oviedo*

# RESUMEN DEL CONTENIDO DE TESIS DOCTORAL

| **1.- Título de la Tesis** | |
|---|---|
| Español/Otro Idioma:<br>Estrategias heurísticas de planificación dinámica de trabajos con límite temporal para la optimización de recursos en entornos de Cloud Computing | Inglés:<br>On-line scheduling of deadline-constrained jobs based on heuristic strategies for the optimization of resources in Cloud Computing environments |

| **2.- Autor** | |
|---|---|
| Nombre:<br>VÍCTOR MANUEL PELÁEZ MARTÍNEZ | DNI/Pasaporte/NIE: |
| Programa de Doctorado: INFORMÁTICA | |
| Órgano responsable: CENTRO INTERNACIONAL DE POSTGRADO | |

**RESUMEN (en español)**


El uso de tecnologías de Cloud híbrido y de servicios de infraestructura pública (*Infrastructures as a Service - IaaS*) permiten a los desarrolladores de servicios ofrecer a sus clientes soluciones con muy poca inversión de antemano y adaptándose a diferentes cargas de trabajo. Las tecnologías de Cloud híbrido permiten complementar las infraestructuras locales (Cloud privado) con recursos computacionales contratados a proveedores de Cloud públicos. El principal problema que se plantea al emplear Clouds híbridos es la minimización de los costes de la infraestructura pública contratada mientras se proporciona la calidad de servicio demandada por el usuario final.

Diversas estrategias de planificación han sido propuestas en el estado del arte para solucionar este problema en aplicaciones que manejan cargas de trabajo de tipo bolsa de tareas y que tienen un límite temporal de ejecución (*deadline-constrained Bag-of-tasks workloads*). Sin embargo, muchas de estas soluciones no consideran la heterogeneidad y variabilidad de rendimiento de los recursos computacionales ofrecidos por los Clouds públicos, el retardo en el aprovisionamiento de nuevas instancias de computación (máquinas virtuales) y la impracticabilidad de tener que disponer de antemano de estimaciones de los tiempos de ejecución de las tareas.

En este trabajo se propone una estrategia de planificación que es capaz de solucionar las limitaciones anteriores y que puede minimizar el coste de la infraestructura contratada mientras se maximiza el número de tareas ejecutadas antes del límite temporal de ejecución. La estrategia se compone de un algoritmo de planificación y de un estimador del tiempo de ejecución de las tareas basado en datos muestrales que es capaz de realizar estimaciones de manera autónoma y que evita que estos tiempos deban ser entradas del algoritmo. Se propone también un estimador basado en la desigualdad de Chebyshev que no realiza asunciones previas acerca de la distribución de los tiempos de ejecución. Además, el planificador tiene en cuenta la heterogeneidad de los Clouds públicos y en su validación se ha considerado el tiempo de aprovisionamiento de las máquinas virtuales.

Para validar el planificador se ha desarrollado un simulador de eventos discretos que ha permitido probar la estrategia con tres cargas de entrada en 8 escenarios en los que varían

aspectos como el límite temporal de los trabajos, la tasa de llegada de aplicaciones o el tiempo de aprovisionamiento de instancias de máquinas virtuales. Cada uno de los escenarios y cargas se ha evaluado contra 8 estrategias de planificación resultado de combinar diferentes algoritmos de planificación y estimadores de los tiempos de procesamiento. Uno de los estimadores y uno de los algoritmos ya existían previamente en el estado del arte y se emplean como base de comparación. Los resultados de la simulación muestran que el algoritmo de planificación propuesto y el estimador de Chebyshev son capaces de obtener resultados similares o mejores en términos de plazos de ejecución cumplidos con respecto a otras técnicas propuestas en el estado del arte.

**RESUMEN (en Inglés)**

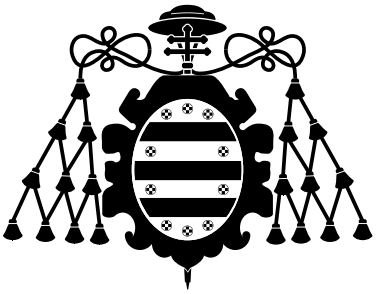
The use of hybrid Cloud technologies and public Infrastructures as a Service (IaaS) allow service developers to offer services to their customers with little upfront investment and to adapt services to different workload sizes. Hybrid Cloud technologies enable the extension of local computing infrastructures (Private Cloud) with computational resources hired from Public Cloud providers.The problem of minimizing the costs of the hired public infrastructure while providing the quality of service needed by the final customer arises when using hybrid clouds.

Several scheduling strategies have been proposed to solve this problem for services dealing with deadline-constrained Bag-of-tasks workloads. Most of these solutions do not consider the heterogeneity and variable performance of the clouds, the provisioning delay of virtual machine instances that affects the elasticity and the impracticality of having good processing time estimations in real systems.

This work proposes a scheduler strategy that overcomes previous limitations and that can minimize the cost of the infrastructure while maximizing the number of deadlines met by the service. The strategy involves an scheduler algorithm and a tasks' runtime estimator based on sampled data that is able to generate estimations autonomously avoiding the need of runtime estimations as inputs of the algorithm. An estimator based on the Chebyshev's inequality it is also proposed. This estimator makes no previous assumptions about the distribution of the runtimes. The solution considers the heterogeneity of Public Clouds and the provisioning time of the virtual machine instances.

A discrete event simulator was developed to validate the scheduler strategy. The strategy was tested using three workloads and 8 scenarios with different deadlines, application arrival rates and provisioning times of the virtual machine instances. Each scenario and workload was evaluated with 8 scheduling strategies resulting from combining different scheduling algorithms and tasks' runtime estimators. One of the estimators and one of the algorithms were already proposed in the state of the art and were used as a comparison baseline. Simulation results show that the scheduler and the Chebyshev based estimator obtains better or similar results in terms of deadlines met than previous techniques in the state of the art.

**SR. PRESIDENTE DE LA COMISIÓN ACADÉMICA DEL PROGRAMA DE DOCTORADO EN INFORMÁTICA**



Universidad de Oviedo
*Universidá d'Uviéu*
*University of Oviedo*
Departamento de Informática

Tesis doctoral

# Estrategias heurísticas de planificación dinámica de trabajos con límite temporal para la optimización de recursos en entornos de Cloud Computing

presentado por
**Víctor Manuel Peláez Martínez**

**Director:** Antonio M. Campos López
**Tutor:** Daniel F. García Martínez

**2019**

# Resumen

El uso de tecnologías de Cloud híbrido y de servicios de infraestructura pública (*Infrastructures as a Service - IaaS*) permiten a los desarrolladores de servicios ofrecer a sus clientes soluciones con muy poca inversión de antemano y adaptándose a diferentes cargas de trabajo. Las tecnologías de Cloud híbrido permiten complementar las infraestructuras locales (Cloud privado) con recursos computacionales contratados a proveedores de Cloud públicos. El principal problema que se plantea al emplear Clouds híbridos es la minimización de los costes de la infraestructura pública contratada mientras se proporciona la calidad de servicio demandada por el usuario final.

Diversas estrategias de planificación han sido propuestas en el estado del arte para solucionar este problema en aplicaciones que manejan cargas de trabajo de tipo bolsa de tareas y que tienen un límite temporal de ejecución (*deadline-constrained Bag-of-tasks workloads*). Sin embargo, muchas de estas soluciones no consideran la heterogeneidad y variabilidad de rendimiento de los recursos computacionales ofrecidos por los Clouds públicos, el retardo en el aprovisionamiento de nuevas instancias de computación (máquinas virtuales) y la impracticabilidad de tener que disponer de antemano de estimaciones de los tiempos de ejecución de las tareas.

En este trabajo se propone una estrategia de planificación que es capaz de solucionar las limitaciones anteriores y que puede minimizar el coste de la infraestructura contratada mientras se maximiza el número de tareas ejecutadas antes del límite temporal de ejecución. La estrategia se compone de un algoritmo de planificación y de un estimador del tiempo de ejecución de las tareas basado en datos muestrales que es capaz de realizar estimaciones de manera autónoma y que evita que estos tiempos deban ser entradas del algoritmo. Se propone

también un estimador basado en la desigualdad de Chebyshev que no realiza asunciones previas acerca de la distribución de los tiempos de ejecución. Además, el planificador tiene en cuenta la heterogeneidad de los Clouds públicos y en su validación se ha considerado el tiempo de aprovisionamiento de las máquinas virtuales.

Para validar el planificador se ha desarrollado un simulador de eventos discretos que ha permitido probar la estrategia con tres cargas de entrada en 8 escenarios en los que varían aspectos como el límite temporal de los trabajos, la tasa de llegada de aplicaciones o el tiempo de aprovisionamiento de instancias de máquinas virtuales. Cada uno de los escenarios y cargas se ha evaluado contra 8 estrategias de planificación resultado de combinar diferentes algoritmos de planificación y estimadores de los tiempos de procesamiento. Uno de los estimadores y uno de los algoritmos ya existían previamente en el estado del arte y se emplean como base de comparación. Los resultados de la simulación muestran que el algoritmo de planificación propuesto y el estimador de Chebyshev son capaces de obtener resultados similares o mejores en términos de plazos de ejecución cumplidos con respecto a otras técnicas propuestas en el estado del arte.

# Abstract

The use of hybrid Cloud technologies and public Infrastructures as a Service (IaaS) allow service developers to offer services to their customers with little upfront investment and to adapt services to different workload sizes. Hybrid Cloud technologies enable the extension of local computing infrastructures (Private Cloud) with computational resources hired from Public Cloud providers.The problem of minimizing the costs of the hired public infrastructure while providing the quality of service needed by the final customer arises when using hybrid clouds.

Several scheduling strategies have been proposed to solve this problem for services dealing with deadline-constrained Bag-of-tasks workloads. Most of these solutions do not consider the heterogeneity and variable performance of the clouds, the provisioning delay of virtual machine instances that affects the elasticity and the impracticality of having good processing time estimations in real systems.

This work proposes a scheduler strategy that overcomes previous limitations and that can minimize the cost of the infrastructure while maximizing the number of deadlines met by the service. The strategy involves an scheduler algorithm and a tasks' runtime estimator based on sampled data that is able to generate estimations autonomously avoiding the need of runtime estimations as inputs of the algorithm. An estimator based on the Chebyshev's inequality it is also proposed. This estimator makes no previous assumptions about the distribution of the runtimes. The solution considers the heterogeneity of Public Clouds and the provisioning time of the virtual machine instances.

A discrete event simulator was developed to validate the scheduler strategy. The strategy was tested using three workloads and 8 scenarios with different

deadlines, application arrival rates and provisioning times of the virtual machine instances. Each scenario and workload was evaluated with 8 scheduling strategies resulting from combining different scheduling algorithms and tasks' runtime estimators. One of the estimators and one of the algorithms were already proposed in the state of the art and were used as a comparison baseline. Simulation results show that the scheduler and the Chebyshev based estimator obtains better or similar results in terms of deadlines met than previous techniques in the state of the art.

# Índice general

# Índice de figuras

# Índice de tablas

# Índice de algoritmos

# 1

# Introducción

## 1.1. Motivación

El paradigma del Cloud Computing [1] ha cambiado la forma en que se emplean los recursos computacionales desde el punto de vista de su adquisición, instalación y uso [2]. Plataformas como Amazon AWS EC2, Microsoft Azure o Google Compute Engine ofrecen recursos computacionales (CPU, GPU, memoria, almacenamiento, ancho de banda, etc.) bajo el paradigma de infraestructura como servicio (*Infraestructure as a Service*) o por sus siglas en inglés *IaaS*. Las principales características de este tipos de servicios son [3]:

- Recursos gestionados bajo demanda y en un modelo de autoservicio.
- Recursos computacionales virtualizados que son accedidos mediante redes de comunicaciones.
- Agrupación de recursos para su uso compartido (en inglés *pooling*).

- Rápida elasticidad. Se puede aumentar o disminuir el tamaño y número de los recursos de manera dinámica y rápida.
- Pago por uso. Los recursos computacionales se alquilan por periodos de tiempo pagando solo por dicho tiempo.

Las infraestructuras como servicio permiten a los proveedores de servicios el acceso a agrupaciones de recursos bajo el modelo económico de pago por uso (alquiler). En este escenario es muy importante la optimización de los recursos computacionales alquilados mientras el servicio proporciona la calidad de servicio requerida por los usuarios finales. La principal estrategia empleada en el estado del arte para esta optimización de los recursos computacionales pasa por la ejecución de las cargas de trabajo mediante un planificador de trabajos que al mismo tiempo gestione la infraestructura alquilada [4].

En el caso de servicios con cargas de trabajo formadas por trabajos con un límite temporal de finalización (*deadline*) la calidad de servicio viene determinada principalmente por el límite temporal de finalización de los trabajos. En este caso el problema para el proveedor de servicios es finalizar la ejecución de la carga de trabajo proporcionada por los usuarios antes del límite temporal que éstos establezcan, al mismo tiempo que se minimiza el coste de la infraestructura contratada al proveedor IaaS para poder ejecutar dicha carga. Este problema es similar a los problemas de planificación de trabajos con límite temporal cuando se tiene un número dinámico de máquinas heterogéneas para la ejecución de los trabajos. El número de tipos de máquinas está relacionado con el número de tipos de máquinas virtuales proporcionadas por los proveedores IaaS.

Las cargas de trabajo compuestas por un conjunto de trabajos o tareas independientes se conocen comúnmente como bolsas de tareas o en inglés *Bag-of-Tasks (BoT)* [5]. Este tipo de cargas de trabajo son muy relevantes dado que muchas aplicaciones reales pueden ser modeladas como bolsas de tareas independientes:

- Simulaciones de Monte Carlo
- Barrido paramétrico
- Cálculo de fractales
- Biología computacional
- Imagen por computador
- Algoritmos de minería de datos

- Búsquedas masivas (key breaking)

Este tipo de cargas de trabajo son comunes en campos de la ciencia como la astronomía, física de altas energías o bioinformática entre otras y son empleados tanto por la comunidad científica como por empresas [6]. Cabe destacar que las cargas de trabajo de tipo BoT suponen el 75 % de las cargas totales presentes en la mayoría de las trazas de sistemas distribuidos reales como LCG, Grid5000, DAS y NorduGrid y son responsables del 90 % del tiempo total de CPU consumido [7].

Este trabajo aborda la optimización de los recursos computacionales de una infraestructura Cloud híbrida para la ejecución de cargas de trabajo formadas por bolsas de tareas con un límite temporal de finalización. El objetivo es maximizar el número de bolsas de tareas que son ejecutadas antes de su límite de finalización mientras que se minimiza el coste de la infraestructura Cloud necesaria para ello.

La principal estrategia para la optimización de los recursos en un entorno Cloud es la correcta gestión de los recursos computacionales contratados en el Cloud público y la ejecución de las aplicaciones empleando un planificador [4]. Según el trabajo de Singh et al [8] los principales problemas que se plantean a la hora de la planificación de recursos son la heterogeneidad y la incertidumbre de los recursos. La heterogeneidad hace referencia a la variabilidad en el número de tipos heterogéneos de máquinas virtuales disponibles en los Cloud públicos. La incertidumbre hace referencia a la variabilidad en el rendimiento de los diferentes recursos como el ancho de banda o la velocidad de la CPU.

## 1.2. Aportaciones

Los trabajos previos en cuanto a planificación de bolsas de tareas con límite temporal (*deadline-constrained BoT workloads*) en el contexto de la computación en el Cloud incluyen diversas técnicas heurísticas para el control de admisión y la planificación de trabajos. El análisis del estado del arte realizado por Thai et al [9] presenta cuatro problemas aún sin resolver en relación a las cargas de trabajo de tipo bolsa de tareas:

1. Variaciones de rendimiento producidas por la heterogeneidad de la infraestructura hardware subyacente a los proveedores de Cloud públicos.

2. Planificación combinando diferentes plataformas y proveedores Cloud.

3. Predicción de los parámetros de rendimiento como por ejemplo la estimación de los tiempos de ejecución de las tareas.

4. Optimización del uso de instancias de computación que están reservadas de antemano y que son de uso exclusivo para una aplicación o servicio durante un tiempo fijado con antelación.

Al contrario que muchos otros trabajos del estado del arte, este trabajo aborda específicamente dos de los cuatro puntos sin resolver listados anteriormente: la heterogeneidad de las infraestructuras (punto 1) y la predicción de los parámetros (punto 3). Con respecto a la heterogeneidad de las infraestructuras muchos trabajos no tienen en cuenta las diferentes velocidades de procesamiento de las instancias de máquinas virtuales ofrecidas por lo proveedores, así como el tiempo de aprovisionamiento necesario para poner en funcionamiento una nueva instancia. Otra limitación importante está relacionada con la predicción de parámetros y es que muchos trabajos asumen que el usuario final de un servicio es capaz de proporcionar una estimación del tiempo de procesamiento de su carga de trabajo cuando ésta es enviada a un servicio para su ejecución. En los trabajos previos analizados la estimación del tiempo de ejecución de las tareas es siempre un parámetro de entrada al planificador, lo cual resulta muy problemático dado que: (1) la estimación precisa de los tiempos de ejecución de las tareas de diferentes usuarios es una tarea compleja [10] y (2) los entornos Cloud no presentan un rendimiento regular [11] [12]. Aunque las plataformas de Cloud públicas comerciales tienen periodos de tiempo con un rendimiento estable y homogéneo, la mitad de los servicios Cloud investigados en el trabajo de Iosup et al [13] presentaron variaciones de rendimiento en forma de patrones de rendimiento repetitivos en ciclos anuales o diarios. Los resultados del trabajo de Schad et al [12] indican que la elección de uno u otro centro de datos dentro de un mismo proveedor de Cloud influye en las variaciones de rendimiento, siendo las diferentes más evidentes para aquellos casos en los que se cuenta con instancias de máquinas virtuales con diferentes velocidades de procesamiento.

La principal contribución de este trabajo sobre el estado del arte es un algoritmo de planificación que considera diferentes velocidades de procesamiento y tiempos de aprovisionamiento de las instancias de máquinas virtuales y que es capaz de generar de manera online y autónoma estimaciones de los tiempos de procesamiento de las tareas en base a las medidas de los tiempos de procesamiento de tareas ejecutadas previamente. Esta estrategia permite al sistema trabajar de manera autónoma y evitar la necesidad de que los usuarios proporcionen estimaciones de los tiempos de procesamiento como entradas. El sistema es además capaz de adaptarse a los cambios en las cargas de trabajo o en la

variaciones de rendimiento de las instancias de máquinas virtuales a lo largo del tiempo.

## 1.3. Estructura del documento

Esta tesis doctoral esta estructurada como sigue. El siguiente capítulo presenta los trabajos relacionados y el análisis del estado del arte. El capítulo 3 describe el modelado del problema objeto de estudio. El capítulo 4 presenta la estrategia de planificación compuesta por un algoritmo de planificación y un estimador de los tiempos de procesamiento de las tareas. El capítulo 5 presenta la evaluación del algoritmo propuesto con diferentes tipos de cargas de trabajo de entrada y en diferentes escenarios. Finalmente el capítulo 6 presenta las conclusiones y trabajos futuros.

# 2

# Trabajo relacionado

Este capítulo describe el estado del arte relacionado con el trabajo en tres secciones: la primera sección describe los conceptos básicos del paradigma de computación en el Cloud, la segunda sección está dedicada a la planificación de tareas con límite temporal de ejecución y la tercera a la estimación de los tiempos de ejecución de tareas.

## 2.1. El paradigma del Cloud Computing

Según la definición del NIST (*National Institute of Standards and Technology of the U.S. Department of Commerce*) [1], la computación en Cloud es un modelo que habilita un acceso por red ubicuo, práctico y bajo demanda a un grupo de recursos computacionales configurables (e.g. redes, servidores, almacenamiento, aplicaciones y servicios) y que pueden ser rápidamente aprovisionados y liberados con un esfuerzo de gestión o una interacción con el proveedor del servicio mínima. Este modelo de computación fomenta la alta disponibilidad y se asocia a cinco características esenciales, tres modelos de servicio y cuatro

modelos de utilización.

Las características esenciales del modelo de computación Cloud son:

- Autoservicio bajo demanda. El usuario puede aprovisionar recursos computacionales, como tiempo de procesamiento o espacio de almacenamiento, según necesite y de manera automatizada sin requerir intervención humana por parte del proveedor del servicio.

- Acceso con redes de banda ancha. Los recursos están disponibles a través de la red y son accedidos mediante mecanismos estándar que promueven el uso de plataformas cliente ligeras o pesadas (e.g. móviles, ordenadores portátiles).

- Agrupación de recursos. Los recursos del proveedor de servicios son agrupados para servir a múltiples clientes en un modelo multi-cliente (en inglés *multitenant*), con diferentes recursos físicos y virtuales dinámicamente asignados y reasignados de acuerdo a la demanda del cliente. Algunos ejemplos de recursos son el almacenamiento, procesamiento, memoria, ancho de banda y máquinas virtuales.

- Elasticidad rápida. Los recursos pueden ser aprovisionados rápida y elásticamente, en algunos casos de manera automática, para responder de manera ágil a la demanda del cliente. Desde el punto de vista del cliente los recursos disponibles parecen muchas veces ilimitados y pueden ser adquiridos en cualquier cantidad y en cualquier momento.

- Servicios monitorizados. Los sistemas Cloud controlan y optimizan automáticamente el uso de los recursos mediante sistemas de monitorización. El uso de los recursos puede ser monitorizado, controlado y consultado, proporcionando transparencia para el proveedor y el cliente acerca de los recursos utilizados.

Los tres modelos de servicio de la computación en Cloud son:

- Software en Cloud como servicio (en inglés *Software as a Service - SaaS*). El proveedor ofrece al cliente el uso de una aplicación ejecutándose en una infraestructura Cloud. Esta aplicación suele estar accesible desde varios dispositivos clientes a través de una aplicación Web. El cliente no gestiona o controla la infraestructura Cloud subyacente (redes, servidores, sistemas operativos, etc.).

- Plataformas Cloud como servicio (en inglés *Platform as a Service - PaaS*). El proveedor ofrece al cliente la posibilidad de instalar o desplegar en su infraestructura Cloud aplicaciones creadas por el cliente empleando lenguajes de programación y herramientas soportadas por el proveedor. Como en el caso anterior, el cliente no gestiona o controla la infraestructura Cloud subyacente (redes, servidores, sistemas operativos, etc.), pero tiene control sobre las aplicaciones desplegadas y sobre algunas configuraciones del entorno de la aplicación.

- Infraestructura Cloud como servicio (en inglés *Infrastructure as a Service - IaaS*). El proveedor ofrece al cliente la posibilidad de aprovisionar almacenamiento, redes, capacidad de procesamiento y cualquier otro recurso computacional que le permita ejecutar cualquier tipo de software, incluyendo sistemas operativos y aplicaciones. El cliente no controla o gestiona la infraestructura física subyacente pero tiene control sobre los sistemas operativos, almacenamiento, aplicaciones y ciertos aspectos de la red.

Los cuatro modelos de utilización de la computación en Cloud son:

- Cloud privado. La infraestructura Cloud es operada para una sola organización cliente. Esta infraestructura puede ser gestionada por esa misma organización o por una tercera entidad y puede residir en las instalaciones de la organización cliente o en las instalaciones de un tercero.

- Cloud comunitario. La infraestructura Cloud es compartida por varias organizaciones con algún tipo de vínculo o interés común.

- Cloud público. La infraestructura Cloud está disponible para el público en general o para un amplio grupo de clientes y es propiedad de un organización que comercializa los servicios Cloud.

- Cloud híbrido. La infraestructura Cloud esta compuesta por dos o más tipos de Clouds (privados, comunitarios o públicos) que son empleados como uno solo mediante el uso de tecnologías que permiten la portabilidad de datos y aplicaciones entre los recursos de los diferentes Clouds.

Este trabajo esta orientado al uso de Infraestructuras Cloud como servicio en modelos de Cloud híbrido, estando el planificador pensado para aquellas organizaciones clientes que quieren escalar los recursos computacionales aprovisionados para ejecutar aplicaciones con cargas de trabajo variables.

## 2.2. Planificación de bolsas de tareas con límite temporal de ejecución

El paradigma de la computación en el Cloud es uno de los paradigmas de computación más importantes hoy en día[4], motivando una extensa actividad investigadora en torno a la problemática de la planificación de trabajos bajo este paradigma. Los trabajos de investigación en este área se pueden clasificar en función de cuatro aspectos diferentes:

1. El tipo de carga de trabajo según la estructura de la misma (trabajos independientes, grupos de trabajos dependientes, grupos de trabajos secuenciales, etc.) y según las restricciones que se aplican a su ejecución (trabajos con límite temporal de ejecución, necesidades de memoria, etc.)

2. El tipo de infraestructura Cloud empleada: Cloud privado, Cloud público o Cloud híbrido.

3. El objetivo del algoritmo de planificación: minimización del consumo energético, optimización de costes económicos, maximización del acuerdo de nivel de servicio (en inglés *Service Level Agrement*, *SLA*).

4. La tipología del algoritmo de planificación: offline, si todos los trabajos a planificar se conocen cuando se inicia la planificación, y online, cuando los trabajos a planificar van llegando al sistema en instantes de tiempo desconocidos de antemano [14].

Este trabajo se centra en la planificación de trabajos independientes o con dependencias muy bajas, que son agrupados en conjuntos (bolsas de tareas, del inglés *bag of tasks*, $BoT$) y que tienen un límite temporal para su ejecución (en inglés *deadline-constrained* $BoT$). El planificador propuesto trabaja en entornos de Cloud híbridos y con el objetivo de minimizar el coste de la infraestructura al mismo tiempo que se maximiza el número de bolsas de tareas ejecutadas antes de su límite temporal.

Esta sección presenta trabajos del estado del arte que comparten el mismo objetivo de investigación o uno muy similar.

Van den Bossche et al [15] presentan un algoritmo online de planificación y gestión de instancias de máquinas virtuales para cargas de trabajo compuestas por bolsas de tareas con límite temporal de ejecución en entornos de Cloud híbridos. El algoritmo trata de maximizar el número de bolsas de tareas ejecutadas antes de su límite temporal mientras que se minimiza el coste de la

infraestructura. El algoritmo se basa en la priorización de los trabajos en colas y esta compuesto por dos sub-algoritmos, uno para la parte privada del Cloud híbrido y otro para la parte pública del mismo. Los autores presentan diferentes variaciones del algoritmo en función de la política de ordenación de los trabajos en las colas y de la estrategia empleada para enviar los trabajos al Cloud público. Las políticas empleadas para la ordenación de los trabajos son *Earliest Deadline First (EDF)* y *First Come First Served (FCFS)*. En el Cloud privado los trabajos son asignados a la instancia de máquina virtual en la que se produzca el menor incremento de carga. Para el envío de un trabajo al Cloud público se emplean dos reglas diferentes: enviar la tarea que no puede ejecutarse antes del límite temporal al proveedor de Cloud público más barato o enviar la tarea predecesora más barata al Cloud público. La tarea predecesora más barata se selecciona de entre las tareas anteriores en la cola de trabajos, siendo la que incurre en menores costes de ejecución en el Cloud público. La combinación de las políticas de ordenación de los trabajos junto con las dos reglas de envío de los trabajos al Cloud público dan lugar a cuatro variantes del algoritmo que son evaluadas junto con otro algoritmo orientado solo a costes (sin tener en cuenta el límite temporal de ejecución de los trabajos). La validación del algoritmo se lleva a cabo mediante la simulación de diferentes escenarios. Los autores simulan diferentes grados de error en las estimaciones de los tiempos de ejecución de las tareas que se deben proporcionar como entrada al algoritmo. El algoritmo propuesto no considera el tiempo necesario para el aprovisionamiento de nuevas instancias de máquinas virtuales en el Cloud público y se asume que todas las máquinas virtuales de todos los proveedores de Cloud tienen una velocidad de procesamiento idéntica. Este es uno de los trabajos más completos del estado del arte y por eso será empleado como base de comparación en este trabajo.

Lin et al [16] presentan un algoritmo de planificación online para la gestión de flujos de trabajo (en inglés *workflows*) científicos con limite temporal de ejecución en entornos de Cloud híbrido. Un *workflow* científico es una carga de trabajo formada por bolsas de tareas que se estructuran formando grafos directos acíclicos (del inglés *Directed Acyclic Graphs - DAG*) de manera que las tareas tienen restricciones de orden para su ejecución y una tarea no puede ser ejecutada si sus predecesoras no han finalizado. Las restricciones de orden en la ejecución de las tareas conforman los caminos de ejecución que representan los caminos dentro del grafo del *workflow*. El algoritmo de Lin et al trata de maximizar el número de aplicaciones (*workflows*) ejecutados con éxito antes de su límite temporal mientras se minimiza el coste del Cloud público. El algoritmo se estructura en dos etapas:

1. Para cada tarea se calcula el tiempo máximo de espera, el tiempo más temprano de inicio y el tiempo de finalización más tardío de acuerdo a la estructura del *workflow* y la posición de la tarea dentro de dicho *workflow*.
2. La información calculada en el punto anterior es empleada por un planificador compuesto por un sub-planificador para el Cloud privado y otro para el Cloud público.

Los *workflows* se ordenan en una cola mediante cuatro políticas diferentes: *FCFS*, *EDF*, *Highest Cost First (HCF)* y *Minimum Load Longest Application First (MLF)*. El sub-planificador del Cloud privado ejecuta los *workflows* en el Cloud privado de acuerdo a su orden en la cola y escanea continuamente la cola para enviar aquellos *workflows* que no van a cumplir su límite temporal de ejecución al Cloud público. El envío de los *workflows* que no van a cumplir su límite temporal se produce de dos formas: envío directo (se envía directamente el *workflow* que no va a cumplir) y envío indirecto (se envían al Cloud público aquellos *workflows* que se encuentran primero en la cola pero que cuya ejecución es más barata que la del *workflow* que no va a cumplir). El sub-planificador del Cloud público ejecuta los *workflows* en las instancias de máquinas virtuales más baratas que permitan cumplir el límite temporal. Los resultados muestran que los mejores resultados se obtienen combinando la política *MLF* de ordenación de los trabajos y el envío indirecto de *workflows* al Cloud público. La solución propuesta no contempla el tiempo de aprovisionamiento de nuevas instancias de máquinas virtuales en el Cloud público y tanto los tiempos de procesamiento como la carga inducida por cada tarea deben ser proporcionados como entradas del algoritmo.

Wang et al [17] proponen un planificador para cargas de trabajo compuestas por bolsas de tareas en entornos Cloud híbridos. El problema de planificación se modela como el problema de la mochila multidimensional con un objetivo dual. Las tareas son planificadas en el Cloud privado empleando la regla Max-Min: las tareas en la carga de trabajo más larga son enviadas a las instancias de máquinas virtuales más rápidas para tratar de minimizar el tiempo de finalización. Los tiempos de finalización de las tareas en el Cloud privado se comparan con su límite temporal de ejecución para enviar aquellas tareas que no son viables en el Cloud privado al Cloud público, seleccionado la instancia de máquina virtual más barata para cumplir el límite temporal. Como en los trabajos analizados anteriormente, la estimación del tiempo de procesamiento de las tareas es una entrada necesaria al planificador y la simulación presentada por los autores no considera errores en dicha estimación. La solución se compara contra un planificador offline basado en programación lineal binaria.

Celaya et al [18] proponen un algoritmo para la *planificación justa* (del inglés *fair scheduling*) de cargas de trabajo formadas por bolsas de tareas con límite temporal de ejecución en entornos de computación genéricos (no específicamente Cloud). El objetivo es minimizar la ralentización en la ejecución de las tareas en un entorno de computación compartido pero sin tener en cuenta el coste de la infraestructura. El algoritmo de planificación se basa en un enfoque descentralizado en el que los trabajos se ordenan en una cola mediante la política (*Fair Share Policy - FSP*) propuesta por los propios autores. El tamaño de las tareas, definido en millones de operaciones de punto flotante por segundo (*FLOPS*, en inglés *floating point operations per second*), es uno de los parámetros de entrada del algoritmo.

Moschakis et al [19] presentan un sistema de planificación en modo *batch* para bolsas de tareas sin límite temporal de ejecución basado en un planificador con dos niveles de gestión de tareas. El modelo propuesto no tiene en cuenta la elasticidad de los entornos Cloud dado que se asume un número constante y fijo de instancias de máquinas virtuales en cada Cloud. El sistema funciona enviando las bolsas de tareas a ejecutar al Cloud con menor carga de trabajo (política *Least Loaded Cloud First*). Dentro de cada Cloud, las bolsas de tareas son ordenadas en una cola y planificadas de acuerdo a tres algoritmos diferentes: *Simulated Annealing*, *Tabu Search* and *Fastest processor Larger Task*. El objetivo del planificador es minimizar el tiempo total de ejecución (*makespan*) y el coste. El tiempo de procesamiento de cada tarea se necesita como entrada al planificador. El sistema se evalúa mediante un simulador de eventos discretos obteniéndose los mejores resultados en términos de tiempo medio total de ejecución y eficiencia del coste con *Tabu Search*.

Cai et al [20] proponen un planificador online para cargas de trabajo formadas por *workflows* científicos y con tiempos de ejecución de tareas estocásticos (debido a la incertidumbre en el rendimiento de las máquinas virtuales y a las propiedades de las tareas). El planificador considera límites temporales de ejecución, diferentes factores de error en los tiempos de procesamiento, diferentes tipos de instancias de máquinas virtuales y tiempos de aprovisionamiento de estas instancias. El algoritmo es un planificador dinámico basado en retardo (del inglés *delay-based dynamic scheduler - DDS*) que emplea la esperanza y la varianza de los tiempos de procesamiento de las tareas como estimador del tiempo de procesamiento. El procedimiento para obtener esa esperanza y la varianza no se explica dentro del trabajo. El algoritmo propuesto se compara con dos planificadores conocidos en el estado del arte para las cargas de tipo *workflow* y una variante del algoritmo presentado por los autores: *URH (Unit-aware Rules based Heuris- tic)* [21], *MOHEFET (Multi-Objective Heterogeneous Ear-*

*liest Finish Time)* [22] y *DDS_MAX (DDS based on maximum task execution times)*. Los resultados obtenidos mediante simulación muestran que el algoritmo *DDS* obtiene mejor rendimiento que el resto de planificadores en términos de tareas ejecutadas antes del límite temporal, pero no en términos de coste para algunos escenarios con límites temporales muy ajustados. Los autores indican que este sobrecoste se debe a la sobreestimación del estimador propuesto y que son necesarios otros tipos de estimadores mas apropiados. Como se verá más adelante nuestro planificador también contempla tiempos de procesamiento estocásticos y el método de estimación propuesto se encuentra integrado con el propio planificador, pudiendo además controlar la sobreestimación de los tiempos de procesamiento.

Oprescu et al [23] presenta un planificador offline para bolsas de tareas con restricciones en el coste de ejecución. El objetivo del planificador es minimizar el tiempo total de ejecución (en inglés *makespan*) cumpliendo la restricción de coste. La planificación tiene dos fases: muestreo y ejecución. Durante la fase de muestreo, un pequeño subconjunto de las tareas es ejecutado en diferentes instancias de máquinas virtuales de diferentes proveedores Cloud. Esto permite al sistema obtener estimaciones de la media de los tiempos de procesamiento para cada tipo de máquina virtual. La estimación se obtiene mediante una simple media móvil. Para disminuir el coste de esta fase de muestreo los autores asumen que el tiempo de procesamiento de las tareas en diferentes tipos de máquinas virtuales exhibe una dependencia lineal y emplean regresión lineal para producir estimaciones en todos los tipos de máquinas virtuales a partir del muestreo. La planificación se realiza con las estimaciones de los tiempos de procesamiento mediante la resolución de un problema de programación entera no lineal. El problema se resuelve mediante programación dinámica como una modificación del problema de la mochila 0-1 . La fase de ejecución ejecuta la planificación generada y la monitoriza para aplicar correcciones cuando se detectan determinadas condiciones. El sistema es validado mediante una implementación real y empleando una carga de trabajo de 1000 tareas con tiempos de procesamiento siguiendo una distribución normal y una varianza de un tercio (0,33) de la media. Esta carga de trabajo es relativamente pequeña para un entorno real, probablemente debido a la complejidad temporal de resolver el problema de programación lineal. Los resultados muestran que el sistema propuesto es capaz de estimar de manera casi perfecta las propiedades de las tareas, aunque no se indica el efecto de la distribución de los tiempos de procesamiento empleados, y cuya varianza es más pequeña que la empleada habitualmente en otros trabajos (0,50 veces la media).

Arabnejad et al [24] presentan un algoritmo de planificación para cargas de

trabajo de tipo *workflow*. Las tareas son priorizadas por el tamaño del camino más largo desde la tarea a la tarea de finalización del camino en el grafo que representa el *workflow*. La selección del procesador para ejecutar la tarea se realiza teniendo en cuenta el límite temporal de ejecución de la tarea y el coste del procesador. El planificador es evaluado empleando SimGrid y comparado con otros algoritmos del estado del arte mediante cargas de trabajo de entre 30 y 70 tareas tanto generadas aleatoriamente como reales. El algoritmo propuesto obtiene resultados similares a otros algoritmos de alta complejidad como *DCA (Dynamic Constraint Algorithm)* [25], *LOSS1* [26] y *GA (Genetic Algorithm)* [27] pero con la complejidad temporal de algoritmos basados en heurísticos como *BHEFT (Budget-constrained Heterogeneous Earliest Finish Time)* [28] y *RANDOM* (se asigna cada tarea a un procesador de manera totalmente aleatoria). El trabajo parece no tener en cuenta las características específicas de los entornos Cloud como la elasticidad (no se realiza el aprovisionamiento de instancias de máquinas virtuales) y los tiempos de procesamiento de las tareas son una entrada necesaria al planificador.

Moschakis y Karatza [29] comparan tres políticas de planificación en un entorno de Cloud público con una carga de trabajo de tipo banda de tareas (del inglés *gang of tasks*): *Adaptive First Come First Serve (AFCFS)*, *First Come First Serve (FCFS)* y *Largest Job First Served (LJFS)*. Una carga de trabajo del tipo banda de tareas está compuesta por un conjunto de lotes con diferentes trabajos (tareas). Todos los trabajos en el mismo lote deben ser planificadores en diferentes procesadores tratando de garantizar que cada trabajo termina al mismo tiempo que el resto de trabajos del lote. El algoritmo propuesto se basa en una cola primaria para las bandas ordenada mediante la política *FCFS* y un segundo conjunto de colas, una por cada instancia de máquina virtual, ordenadas según la política *AFCFS* o *LJFS*. Dado que no existe restricción temporal o plazo de ejecución, el algoritmo no necesita ninguna estimación de los tiempos de procesamiento. La validación llevada a cabo no contempla variaciones en el rendimiento de las instancias de máquinas virtuales ya que todas las instancias son idénticas. En otro trabajo de los mismos autores [30] se presenta un planificador para cargas de trabajo de tipo bolsas de tareas sin límite temporal de ejecución en entornos Cloud interconectados. Aunque los autores sí modelan en este trabajo diferentes infraestructuras Cloud, tampoco se hace referencia a la variación de rendimiento de diferentes instancias del mismo tipo de máquina virtual o a la variación de los tiempos de procesamiento de los trabajos.

Wu et al [31] presentan un algoritmo de planificación y control de admisión de trabajos para cargas de trabajo formadas por trabajos independientes y con un plazo temporal de ejecución. Esta carga de trabajo podría asemejarse

a bolsas de tareas donde cada bolsa contiene una única tarea. El trabajo considera diferentes proveedores de Cloud público y diferentes tipos de máquinas virtuales con diferentes rendimientos entre los proveedores. El algoritmo trata de maximizar la calidad de servicio al mismo tiempo que minimiza los costes mediante la priorización de los trabajos en colas empleando diferentes políticas de ordenación. La estrategia de admisión de trabajo sigue cuatro políticas diferentes: iniciar una nueva instancia de máquina virtual por cada petición de trabajo, encolar la petición, priorizar las peticiones y retardar las peticiones. El algoritmo de planificación necesita como entrada las estimaciones de los tiempos de procesamiento de los trabajos. Los autores no simulan ningún tipo de error en dichas estimaciones, aunque sí que simulan la variación en el rendimiento de las instancias de máquinas virtuales. Esta variación se simula con una distribución normal que toma como media el valor exacto del parámetro que define el rendimiento y como desviación entre 0 y 0,5 veces el valor de la media.

Calheiros y Buyya [32] presentan un algoritmo similar para el aprovisionamiento de instancias de máquinas virtuales y la planificación de trabajos en Clouds híbridos que manejan cargas de trabajo de tipo bolsas de tareas con límite temporal de ejecución. El algoritmo propuesto agrupa los trabajos según su prioridad y emplea una cola por cada instancia de máquina virtual y nivel de prioridad. Las colas son gestionadas mediante la política *Heterogeneous Earliest Finish Time (HEFT)*. El trabajo asume que existe un único tipo de máquina virtual con el mismo rendimiento en toda la infraestructura Cloud y el algoritmo necesita las estimaciones de los tiempos de procesamiento de los trabajos como entrada. Los autores no consideran ningún tipo de error en las estimaciones o variaciones en el rendimiento de los recursos computacionales.

Como se muestra en la 2.1 casi todos los algoritmos de planificación propuestos son variaciones del conocido algoritmo LIST propuesto originalmente por Graham en 1969 [33]. El algoritmo LIST original funciona de la siguiente manera: dada una permutación (una lista) de un conjunto de trabajos, el siguiente trabajo de la lista se planifica en la primera máquina que este disponible. Aunque simple, el algoritmo ha dado lugar a múltiples variaciones mediante la modificación de dos aspectos [34]:

1. El criterio para ordenar el conjunto de trabajos a planificar en una lista.
2. El criterio para seleccionar el procesador que atenderá la ejecución del trabajo al frente de la lista.

En problemas de planificación complejos donde las soluciones óptimas no son viables por su alta complejidad temporal, el criterio de ordenación de trabajos

o el de selección de procesadores del algoritmo LIST suele basarse generalmente en funciones heurísticas [35].

El estudio del estado del arte para aquellos problemas de planificación de trabajos con límite temporal de ejecución en entornos de computación en Cloud nos ha permitido generalizar el algoritmo LIST en cuatro fases presentes en la casi totalidad de los trabajos:

1. Priorización de trabajos (política de priorización)
2. Identificación de los trabajos que requieren una nueva instancia de máquina virtual para su ejecución (función heurística de tiempo de finalización).
3. Selección de la instancia de máquina virtual ya existente para ejecutar los trabajos (función heurística de coste de ejecución)
4. Selección del tipo de máquina virtual para crear una nueva instancia para aquellos trabajos que no pueden ser ejecutadas en las instancias de máquinas virtuales ya existentes (función heurística de coste de ejecución).

Esta estructura de algoritmo en cuatro fases es el algoritmo de planificación más empleado en el estado del arte y el empleado en este trabajo. Las diferentes entre los diferentes algoritmos residen en la política de priorización de trabajos y en las funciones heurísticas para estimar costes y tiempos de finalización).

Las tablas 2.1 y 2.2 resumen las características de los diferentes trabajos del estado del arte.

Las abreviaturas empleadas en la tabla 2.1 son las siguientes:

- WDT: *Workflow of dependent tasks*
- BoTW: *BoT Workflow*
- DAG: *Direct Acyclic Graph Workflow*
- PART: particionamiento de aplicaciones
- HEU: heurístico.
- NLIP: *Non Linear Integer Program*

Las notas en la tabla 2.1 son las siguientes:

1. Parece un planificador online pero las simulaciones están basadas en un conjunto fijo de trabajos que llegan al mismo tiempo.

| | Carga de trabajo | Límite Temporal | Objetivo | Tipo Cloud | Online/Offline | Tipo algoritmo |
|---|---|---|---|---|---|---|
| Bossche et al [15] | BoT | Si | Max deadline<br>Min cost | Híb | On | LIST |
| Lin et al [16] | WDT | Si | Max deadline<br>Min cost | Híb | On | PART + LIST |
| Moschakis et al [19] | BoT | No | Min makespan<br>Min cost | Híb | Off | LIST + HEU |
| Wang et al [17] (1) | BoT | Si | Max deadline<br>Min cost | Híb | Off | LIST |
| Celaya et al [18] (2) | BoT | Si | Min stretch | Pri | On | LIST |
| Cai et al [20] | BoTW | Si | Max deadline<br>Min cost | Púb | On | LIST |
| Oprescu et al [23] | BoT | No | Min makespan<br>limited cost | Púb | Off | NLIP |
| Arabnejad et al [24] | DAG | Si | Makespan $\leq$ Deadline<br>Cost $\leq$ Budget | Híb | Off | LIST |
| Moschakis et al [29] | GoT | No | Min ART (3)<br>Min cost | Púb | On | LIST |
| Wu et al [31] | Tasks | Si | Max SLA<br>Min cost | Púb | On | LIST |
| Calheiros et al [32] | BoT | Si | Max deadline<br>Min cost | Híb | On | LIST |
| Este trabajo | BoT | Si | Max deadline<br>Min cost | Híb | On | LIST |

Tabla 2.1: Resumen del estado del arte: planificador, carga y entorno Cloud

2. No se consideran costes por lo que no hay diferencias entre infraestructuras privadas y públicas. No es específico de sistemas Cloud, puede ser empleado en clusters o grids tradicionales.

3. ART: *Average Response Time*

Las notas en la tabla 2.2 son las siguientes:

1. El tamaño de las tareas en millones de instrucciones es parte del modelo.

2. La simulación no introduce errores en los tiempos de procesamiento, aunque si variaciones para las tareas del mismo usuario. Además, el planificador considera que está trabajando con estimaciones imprecisas.

La principal limitación de todos los trabajos analizados en los párrafos anteriores es que el planificador necesita como dato de entrada una estimación del tiempo de procesamiento de cada trabajo. Aún en el caso de que realizar dicha estimación de antemano fuese viable, muchos trabajos tampoco tienen en cuenta la heterogeneidad de rendimiento en los entornos Cloud y la variabilidad que estos pueden sufrir a lo largo del tiempo en cuanto a velocidades de ejecución y de transferencia de datos [11]. Esto supone que la misma estimación del tiempo de procesamiento es aplicada a instancias de máquinas virtuales de diferentes entornos Cloud y en diferentes instantes temporales, lo que en muchas ocasiones resulta inválido. Algunos autores tratan de solucionar estas limitaciones modelando cierta imprecisión o error en las estimaciones de los tiempos de procesamiento que son introducidas en la planificación [15]. Esta imprecisión o error suele ser modelada mediante un ruido uniforme que se añade a los tiempos de procesamiento exactos de los trabajos o a los parámetros que definen el rendimiento de las instancias de las máquinas virtuales. Este enfoque no considera estacionalidad o tendencias en el error y por tanto su utilidad es cuestionable por lo que en este trabajo se prefiere el uso de estimaciones de los tiempos de procesamiento generadas online y como parte del propio algoritmo de planificación.

El efecto de estimaciones imprecisas de los tiempos de procesamiento de los trabajos y del tiempo de aprovisionamiento de las instancias de máquinas virtuales ha sido estudiado por diversos autores. Malawski et al [36] evaluaron tres algoritmos de planificación diferentes aplicados a cargas de trabajo científicas con límite temporal de ejecución. En los tres planificadores los tiempos de procesamiento de las tareas es una entrada del algoritmo. La evaluación del error en las estimaciones es simple y no afecta al tiempo total de procesamiento pero

| | Estimaciones tiempos procesamiento en entrada | Diferentes tipos de VM | Error estimaciones tiempos procesamiento | Diferentes velocidades procesamiento en VM | Tiempo aprovisionamiento de VM |
|---|---|---|---|---|---|
| Bossche et al [15] | Si | Si | Si | No | No |
| Lin et al [16] | Si | Si | No | Si | No |
| Moschakis et al [19] | Si | Si | No | No | No |
| Wang et al [17] | Si (1) | Si | No | Si | No |
| Celaya et al [18] | Si | Si | No | Si | No |
| Cai et al [20] | Si | Si | Si | Si | Si |
| Oprescu et al [23] | No | Si | Si | Si | No |
| Arabnejad et al [24] | Si | Si | No | Si | No |
| Moschakis et al [29] | No | No | - | No | Si |
| Wu et al [31] | Si | Si | No | Si | Si |
| Calheiros et al [32] | Si | No | No | No | No |
| Este trabajo | No | Si | Si (2) | Si | Si |

Tabla 2.2: Resumen del estado del arte: características

los resultados muestran que las estimaciones imprecisas de los tiempos de procesamiento y los retardos en el aprovisionamiento de las instancias de máquinas virtuales afectan a los tres algoritmos. Shi et al [37] estudiaron un escenario muy similar empleando diferentes tipos de cargas de trabajo y algoritmos de planificación que necesitaban los tiempos de procesamiento de las tareas y los tiempos de aprovisionamiento de las instancias de máquinas virtuales como entradas. Los resultados indican que el rendimiento de los planificadores es ligeramente inferior cuando se emplean tiempos de procesamiento y de aprovisionamiento imprecisos, aunque el error esta limitado a un 20 % de los valores exactos.

## 2.3. Estimación de los tiempos de procesamiento de los trabajos

El impacto de la calidad de las estimaciones de los tiempos de procesamiento de los trabajos en los algoritmos de planificación ha sido estudiado en las áreas de planificación de trabajos y *Grid Computing* demostrándose su importancia. Lee y Snavely [10] descubrieron que incluso aunque se proporcione un incentivo a los usuarios para mejorar las estimaciones de los tiempos de procesamiento de sus trabajos, muchos usuarios son incapaces de hacerlo. AuYoung el al [38] mostraron que cuando se proporciona a un planificador información imprecisa, como pueden ser tiempos de procesamiento erróneos, las políticas de planificación basadas en la utilidad pueden degradarse a la mitad en comparación con otros enfoques tradicionales.

Tsafrir et al [39] estudiaron el impacto de las estimaciones de los tiempos de procesamiento proporcionadas por los usuarios a los planificadores y concluyeron que usando estimadores simples, como una media de los tiempos de procesamiento de los dos últimos trabajos ejecutados, se podían obtener resultados mejores que empleando las estimaciones proporcionadas por los propios usuarios. Aunque la solución propuesta no era específica de arquitecturas Cloud, pone de manifiesto la importancia de utilizar estimaciones de los tiempos de procesamiento basadas en tiempos de procesamiento medidos previamente.

El principal problema de emplear estimaciones de los tiempos de procesamiento como entradas a un algoritmo de planificación es la imprecisión de dichas estimaciones debida no solo al error propio del método de estimación, sino también a la variabilidad en el rendimiento de los Cloud públicos a lo largo del tiempo. Jackson et al [40] estudiaron la variabilidad en el rendimiento de las instancias de máquinas virtuales en el Cloud público concluyendo que el rendi-

miento se ve afectado por el hardware subyacente a dichas instancias. Esto se debe a que los proveedores de Cloud públicos emplean hardware heterogéneo para ejecutar el mismo tipo de máquinas virtuales, de modo que el rendimiento puede variar de una instancia a otra incluso siendo del mismo tipo.

Verboven et al [41] presentan un modelo para estimar los tiempos de procesamiento de los trabajos basados en tiempos de procesamiento vistos anteriormente y en los parámetros empleados para configurar los trabajos con el objetivo de planificar bolsas de tareas en entornos de Grid Computing. Los resultados de las simulaciones llevadas a cabo muestran que las estimaciones mejoran las técnicas de planificación anteriores.

Silva et al [42] presentan una metodología para la estimación online de los tiempos de procesamiento, espacio en disco y consumo de memoria de cargas de trabajo científicas basada en el tamaño de los datos de entrada. En el trabajo de Pietri et al [43] se presenta un modelo para estimar los tiempos de procesamiento de cargas de trabajo científicas en entornos Cloud que es capaz de predecir los tiempos de procesamiento con un error inferior al 20 % para más del 96 % de los experimentos estudiados. El modelo emplea tiempos de procesamiento medidos en ejecuciones anteriores de los trabajos, así como otra información relativa a la estructura de la carga de la trabajo.

## 2.4. Situación de este trabajo respecto al estado del arte

Tras estudiar el estado del arte podemos concluir que aunque algunas de las soluciones propuestas en el área de la planificación de trabajos en entornos de Cloud consideran la variación en el rendimiento de las capacidades computacionales y el error en las estimaciones de los tiempos de procesamiento de los trabajos, este trabajo cuenta con las siguientes características únicas:

- El algoritmo de planificación no necesita como datos de entrada las estimaciones de los tiempos de procesamiento de los trabajos. Los enfoques empleados hasta el momento resultan de difícil aplicabilidad en un entorno real debido al coste de generar de antemano estimaciones de los tiempos de procesamiento. Además, este trabajo demostrará que el uso de estimaciones generadas en tiempo de ejecución, a partir de tiempos de procesamiento medidos en ejecuciones previas y empleando técnicas estadísticas, permite obtener resultados iguales o mejores que otras soluciones en el estado del arte.

- Se consideran instancias de máquinas virtuales con diferentes velocidades de procesamiento.
- Se tiene en cuenta que el aprovisionamiento de nuevas instancias de máquinas virtuales tarda un tiempo y por tanto no es instantáneo.

# 3

# Definición del problema de planificación

## 3.1. Cloud híbrido

Las aplicaciones y servicios implementados sobre infraestructuras de Cloud híbrido ejecutan las cargas de trabajo sobre instancias de máquinas virtuales que son desplegadas en un centro de computación local (Cloud privado) y uno o más centros de computación públicos y compartidos (Cloud público). El objetivo del planificador propuesto en este trabajo es ejecutar los trabajos que envían los usuarios finales del servicio sobre la infraestructura Cloud híbrida. La figura 3.1 muestra la arquitectura general del sistema y como el planificador propuesto se integra con la implementación del servicio.

Desde el punto de vista del planificador propuesto tanto el Cloud privado como el público funcionan de la misma forma, con la única diferencia de que se asume que el Cloud público tiene un conjunto infinito de instancias de máquinas

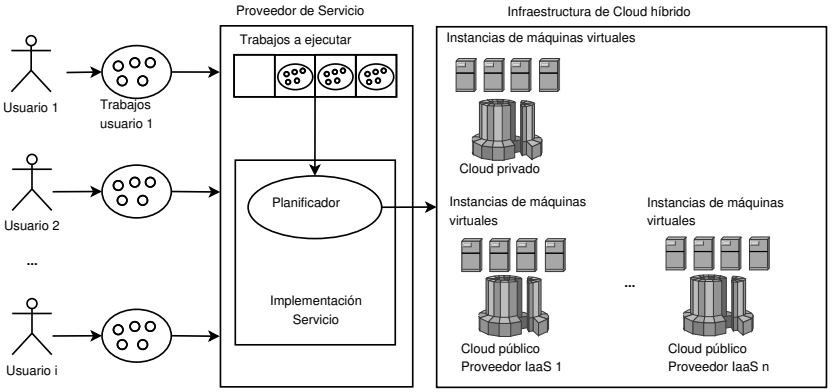



Figura 3.1: Arquitectura general del sistema

virtuales mientras que el Cloud privado puede escalar solo hasta un número limitado de instancias.

En lo referente a los costes, estos son considerados constantes en el Cloud privado, dado que los equipos de computación necesarios están operativos el 100 % del tiempo. Por el contrario, los proveedores IaaS empleados para formar el Cloud público cobran solo por el tiempo de uso de las instancias de máquinas virtuales. Aunque en el pasado el cobro se realizaba por periodos discretos de tiempo, como por ejemplo de hora en hora, en estos momentos los principales proveedores IaaS como Amazon EC2 o Google Compute Engine realizan cobros por tramos de segundo, con un mínimo de 60 segundos por cada instancia. Otros costes facturados por el proveedor son el consumo de ancho banda o el espacio en disco persistente empleado, aunque dichos costes no son tenidos en cuenta en este trabajo al centrarse en cargas de trabajo intensivas en CPU.

Teniendo en cuenta las consideraciones anteriores el modelo de Cloud híbrido empleado en este trabajo se caracteriza de la siguiente manera:

1. El Cloud híbrido se compone por un Cloud privado que reside en la infraestructura local y un Cloud público formado por una o varias infraestructuras públicas proporcionadas por proveedores IaaS.

2. El Cloud privado y el Cloud público tienen un conjunto predefinido de tipos de máquinas virtuales que pueden ser empleados. Cada proveedor IaaS empleado en el Cloud público proporciona diferentes tipos de máquinas virtuales con diferentes características y rendimiento. Las máquinas

virtuales del mismo tipo para el mismo proveedor IaaS podrían también presentar diferente rendimiento.

3. El Cloud privado comienza con un número conocido de instancias de máquinas virtuales que puede incrementarse o decrementarse dinámicamente pero siempre hasta un máximo que depende de las características de la infraestructura hardware subyacente.

4. Se considera que cada proveedor IaaS que forma el Cloud público tiene una capacidad infinita y por tanto el número de instancias de máquinas virtuales que puede proporcionar es infinito.

5. El aprovisionamiento de una nueva instancia de una máquina virtual no es inmediato. Cada proveedor IaaS y cada tipo de máquina virtual puede tener un tiempo de aprovisionamiento diferente que puede variar a lo largo del tiempo.

## 3.2. Carga de trabajo

Cada usuario final del servicio que incluye el planificador envía varias aplicaciones (bolsas de tareas) para su ejecución a diferentes ritmos de llegada. Cada aplicación se compone de un conjunto independiente de tareas o trabajos que pueden ser ejecutadas sin restricciones en la misma o en diferentes instancias de máquinas virtuales ya que no se comunican ni interaccionan entre si. Todas las tareas en la misma aplicación comparten un mismo límite de ejecución temporal (*deadline*). La carga de entrada se define por las siguientes características:

1. Cada usuario envía para su ejecución aplicaciones a diferente ritmo. El ritmo de llegada (*arrival rate*) se modela empleando funciones de distribución de probabilidad para cada usuario. Estas distribuciones de probabilidad son desconocidas para el planificador.

2. El número de tareas en cada aplicación varia de un usuario a otro y también está modelado empleado variables aleatorias con una distribución de probabilidad desconocida para el planificador.

3. El límite temporal de ejecución de cada aplicación es fijado por el usuario y es conocido por el planificador cuando se recibe dicha aplicación.

4. El tiempo de procesamiento de cada tarea de un usuario sigue una distribución de probabilidad que distinta para cada usuario. Obviamente,

el tiempo de procesamiento de una tarea varia dependiendo de las características del hardware subyacente a la instancia de máquina virtual empleada para ejecutarla. Este tiempo de procesamiento es desconocido para el planificador antes de ejecutar la tarea.

5. Cada tarea puede ser ejecutada empleando un solo procesador o núcleo de ejecución en una instancia de máquina virtual. No existe restricción alguna en cuanto a cómo se pueden ejecutar las tareas de una aplicación en diferentes máquinas virtuales o Clouds.

En este trabajo se considera solo el consumo de CPU y no se tiene en cuenta el consumo de tráfico de red o de espacio en disco. Se asume que el coste en tiempo de CPU (el coste en tiempo de instancias de máquinas virtuales) es mucho mayor que el coste de la transferencia de datos de las aplicaciones.

## 3.3. Formalización

Teniendo en cuenta las definiciones de la infraestructura y de la carga de trabajo de las secciones anteriores, el problema formal objeto de este trabajo es la planificación de bolsas de tareas con límite temporal de ejecución tratando de maximizar el número de aplicaciones ejecutadas antes de su límite temporal y minimizando el coste de la infraestructura computacional asociada. Este apartado presenta las definiciones y ecuaciones básicas que formalizan el problema y facilitan la comprensión de los siguientes apartados.

La tabla 3.1 presenta la notación empleada en la formalización de este trabajo. Algunas consideraciones adicionales con respecto a la notación mostrada en la tabla se recogen a continuación:

- Los tipos de máquinas virtuales representan desde el punto de vista del planificador tipos de máquinas no relacionados.
- Las instancias de máquinas virtuales del mismo tiempo de máquina virtual son considerados máquinas idénticas.
- El tamaño de una tarea puede ser medido en millones de instrucciones (MI) u operaciones de punto flotante (FLOPS).
- Cada procesador puede ejecutar tan solo una tarea de cada vez.

Las siguientes definiciones se emplean para formalizar el planificador propuesto.

| Símbolo | Significado |
|---|---|
| $u$ | Índice de un usuario atendido por el sistema |
| $a$ | Índice de una aplicación de tipo bolsa de tareas enviada al sistema por un usuario |
| $j$ | Índice de un trabajo independiente (o tarea) en una aplicación |
| $c$ | Índice de un proveedor de Cloud empleado en el sistema |
| $t$ | Índice de un tipo de máquina virtual |
| $i$ | Índice de una instancia de máquina virtual |
| $k$ | Índice de un procesador en una instancia de máquina virtual |
| $r_{ua}$ | Tiempo de llegada (*release date*) al sistema de la aplicación $a$ del usuario $u$ |
| $l_{uaj}$ | Tamaño de la tarea $j$ en la aplicación $a$ del usuario $u$ |
| $d_{ua}$ | Límite temporal de ejecución (*deadline*) de la aplicación $a$ del usuario $u$ |
| $q_{ct}$ | Número de procesadores en la máquina virtual de tipo $t$ del proveedor Cloud $c$ |
| $v_{ctk}$ | Velocidad de procesamiento del procesador $k$ en la máquina virtual de tipo $t$ del proveedor Cloud $c$ |
| $rqt_{cti}$ | Tiempo de petición de la instancia de máquina virtual $i$ del tipo $t$ y el proveedor Cloud $c$ |
| $avt_{cti}$ | Tiempo de disponibilidad de la instancia de máquina virtual $i$ del tipo $t$ y el proveedor Cloud $c$ |
| $avt_{ctik}$ | Tiempo de disponibilidad del procesador $k$ en la instancia de máquina virtual $i$ del tipo $t$ y el proveedor Cloud $c$ |
| $sdt_{cti}$ | Tiempo de apagado de la instancia de máquina virtual $i$ del tipo $t$ y el proveedor Cloud $c$ |
| $bp_c$ | Periodo de facturación para el proveedor de Cloud $c$ |
| $cvmt_{ct}$ | Coste por periodo de facturación para el tipo de máquina virtual $t$ del proveedor Cloud $c$ |
| $p_{uajctik}$ | Tiempo de procesamiento para el trabajo $j$ en la aplicación $a$ del usuario $u$ en el procesador $k$ de la instancia de máquina virtual $i$ del tipo $t$ procedente del proveedor Cloud $c$ |
| $st_{uajctik}$ | Tiempo de inicio de ejecución del trabajo $j$ en la aplicación $a$ del usuario $u$ en el procesador $k$ de la instancia de máquina virtual $i$ del tipo $t$ procedente del proveedor Cloud $c$ |
| $C_{uajctik}$ | Tiempo de finalización para el trabajo $j$ en la aplicación $a$ del usuario $u$ en el procesador $k$ de la instancia de máquina virtual $i$ del tipo $t$ procedente del proveedor Cloud $c$ |

Tabla 3.1: Símbolos usados en el trabajo

**Definición 1** *Una aplicación o bolsa de tareas de índice a es un conjunto de $J_a$ tareas independientes enviadas para su ejecución por el usuario u. Todos los trabajos tienen el mismo tiempo de llegada, $r_{ua}$, y límite temporal de ejecución, $d_{ua}$, pero pueden tener diferente tamaño, $l_{uaj}$. Los trabajos son independientes y pueden ser ejecutados sin ningún orden o restricción.*

**Definición 2** *El conjunto de todas las aplicaciones enviadas para su ejecución por parte de todos los usuarios se nombra como A.*

**Definición 3** *El conjunto de todos los proveedores de Cloud públicos se denomina $CP$.*

**Definición 4** *El tiempo de puesta en marcha, o aprovisionamiento de una instancia de máquina virtual, se define como $avt_{ctik} - rqt_{ctik}$.*

**Definición 5** *$T_{uaj}$ denota la tardanza (en inglés tardiness) de un trabajo j en la aplicación a del usuario u y se define como $max(C_{uajctik} - d_{ua}, 0)$. Esta métrica se refiere al tiempo de procesamiento de un trabajo más allá de su límite temporal de ejecución.*

**Definición 6** *$cvmi_{cti}$ denota el coste de una instancia de máquina virtual i. Se define como $cvmi_{cti} = ceilling(sdt_{cti} - avt_{cti}, bp_c) \cdot cvmt_{ct}$. Para el Cloud privado $cvmt_{ct} = 0$, i.e. las instancias de máquinas virtuales están siempre operativas. La función ceilling retorna el número inmediatamente superior resultado de dividir $sdt_{cti} - avt_{cti}$ entre $bp_c$. Esto garantiza que el coste de un instancia de máquina virtual sigue el periodo de facturación del proveedor Cloud.*

**Definición 7** *El tiempo de procesamiento de los trabajos se define a partir de su tiempo de inicio y de finalización: $p_{uajctik} = C_{uajctik} - st_{uajctik}$.*

**Definición 8** *Una planificación, S, asigna trabajos a un conjunto de instancias de máquinas virtuales de acuerdo al tiempo de llegada y el límite temporal de ejecución de dichos trabajos. El planificador asigna un tiempo de inicio de ejecución a cada trabajo y este tiempo determina su tiempo de finalización. Se puede definir como un conjunto de tuplas: $S = (u, a, j, c, t, i, k, st_{uajctik,C_{uajctik}}) \forall u, a, j$.*

**Definición 9** *Un planificador es una función $f : (A, CP) \rightarrow S$.*

El problema estudiado en este trabajo se puede definir de acuerdo a la notación propuesta en el trabajo de Graham et al.[44]. De acuerdo a esta notación, cada problema de planificación puede ser representado mediante una notación

de tres campos $\alpha|\beta|\gamma$, donde $\alpha$ especifica el entorno de máquinas, $\beta$ especifica las características de los trabajos y $\gamma$ denota el criterio de optimalidad. De acuerdo a esta notación nuestro problema puede ser definido como:

$$Rm|r_{ua}, noprmp, batch, setup, d_{ua}| \sum {T_{uaj}}^{(1)}, \sum {cvmi_{cti}}^{(2)}$$

- $Rm$: Existen $m$ máquinas en paralelo con diferentes velocidades de procesamiento no relacionadas entre sí.
- $r_{ua}$: Los trabajos tienen tiempos de llegada y su ejecución no puede comenzar antes de ese momento. Todos los trabajos en la misma aplicación comparten el mismo tiempo de llegada.
- *noprmp*: El planificador no puede interrumpir la ejecución de un trabajo una vez que esta comienza, es un planificador sin desalojo (en inglés *non preemptive*).
- *batch*: Los trabajos se agrupan para su ejecución en grupos denominados *bolsas*, cada bolsa de tareas se considera un lote (en inglés *batch*).
- *setup*: Los trabajos se ven afectados por los tiempos de puesta en marcha de las máquinas. El primer trabajo ejecutado en una instancia de máquina virtual debe esperar a que dicha instancia este lista.
- $d_{ua}$: Los trabajos deben ser ejecutados antes de su límite temporal de ejecución. Todos los trabajos en la misma bolsa de tareas tienen un límite común.
- $\sum {T_{uaj}}^{(1)}$: Tardanza total.
- $\sum {cvmi_{cti}}^{(2)}$: Coste total de los recursos empleados en el Cloud público. Esta métrica se refiere al coste de ejecución de los trabajos y está relacionada con sus tiempos de finalización. Se conoce también como el tiempo total ponderado de finalización (en inglés *total weighted completion time*) en la literatura.

A modo de resumen, el objetivo del planificador es producir una planificación que minimice el número de aplicaciones que son ejecutadas con tiempos de finalización posteriores a su límite temporal (objetivo principal) y minimizar el coste de las instancias de máquinas virtuales empleadas para dicha ejecución (objetivo secundario).

Además, el problema de planificación estudiado en este trabajo puede ser clasificado como un problema de planificación online dado que la función de planificación no conoce de antemano cuántos trabajos tiene que procesar ni cuáles son sus tiempos de llegada [45]. Dado que los usuarios del sistema envían aplicaciones para su ejecución de manera continua e imprevisible, no es posible conocer de antemano el número de trabajos que atenderá el sistema. Además, los tiempos de procesamiento o ejecución de los trabajos están sujetos a fluctuaciones, incluso en la misma instancia de máquina virtual, y solo son conocidos cuando el trabajo finaliza su ejecución. El planificador no conoce en ningún instante cuántos trabajos faltan por llegar y cuáles son los instantes de llegada.

# 4

# Estrategia de planificación

Según los estudios realizados en este trabajo nunca ha sido publicada la complejidad del problema de planificación objeto de estudio, ni en su versión mas simple (offline y determinista), ni en su versión mas compleja y estudiada aquí (online y no determinista):

$$Rm|r_{ua}, noprmp, batch, setup, d_{ua}|\sum {T_{uaj}}^{(1)}, \sum {cvmi_{cti}}^{(2)} \tag{4.1}$$

Sin embargo, se sabe que la complejidad de los siguientes problemas, en su versión offline y determinista, es NP duro en sentido estricto [46] [47] [48]:

$$1|r_j|\sum T_j \tag{4.2}$$

$$1||\sum w_j T_j \tag{4.3}$$

Esta complejidad sugiere que el coste computacional de encontrar una solución óptima para el problema objeto de estudio es alto incluso para instancias de problemas con pocas máquinas y trabajos. Además, se sabe que para dos o mas

procesadores ningún algoritmo de planificación basado en límites temporales puede ser óptimo sin conocer de antemano los límites temporales, los tiempos de procesamiento y los instantes de inicio de las tareas [49]. Dado que en el caso objeto de estudio los tiempos de procesamiento de las tareas nunca son conocidos de antemano, se justifica el uso de técnicas heurísticas para encontrar soluciones subóptimas con costes computacionales moderados.

La estrategia más simple para la planificación online consiste en asignar cada trabajo a un procesador tan pronto como el trabajo llega al sistema y un procesador queda libre [45]. Esta estrategia tan simple no es válida en un entorno de Cloud híbrido porque el número de procesadores puede ser modificado dinámicamente y el planificador debe decidir cuándo aprovisionar nuevos procesadores.

El algoritmo de planificación propuesto se basa en transformar el problema de planificación online en un problema semi-offline alimentando el algoritmo con aproximaciones (estimaciones) de los tiempos de procesamiento de las tareas. Entonces, es posible la aplicación de algunas de las estrategias conocidas para los problemas de planificación deterministas y offline del problema similar $1|r_j|\sum T_j$. Con este enfoque el planificador aprovisiona un nuevo procesador solo cuando estima que algún trabajo no va a cumplir su límite temporal de ejecución de acuerdo a la configuración actual de procesadores [46]. El problema continúa siendo semi-online porque el planificador no conoce el número total de trabajos ni sus tiempos de llegada.

Tal y como se muestra en la figura 4.1 el planificador se compone de dos sub-planificadores. Inicialmente todas las aplicaciones se envían al planificador del Cloud privado para su ejecución. Si no existen instancias de máquinas virtuales disponibles para la ejecución de un trabajo dentro de su límite temporal, entonces este trabajo y todos los trabajos sin ejecutar en su misma bolsa de tareas son enviados al planificador del Cloud público. Para minimizar el coste de la infraestructura de Cloud público el planificador del Cloud público calcula el tipo de máquina virtual más barata para ejecutar cada trabajo antes de su límite temporal.

El planificador asigna a cada trabajo $j$, dentro de la aplicación $a$ del usuario $u$, un procesador $k$ de la instancia de máquina virtual $i$ del tipo de máquina virtual $t$ del proveedor de Cloud público $c$, calculando un tiempo de inicio de ejecución estimado $est_{uajctik}$ y un tiempo de finalización estimado $ec_{uajctik}$.

El tiempo de finalización estimado se puede expresar como la suma del tiempo esperado de inicio de ejecución y el tiempo de procesamiento estimado para el trabajo ($ep_{uajctik}$) de acuerdo a la ecuación 4.4.

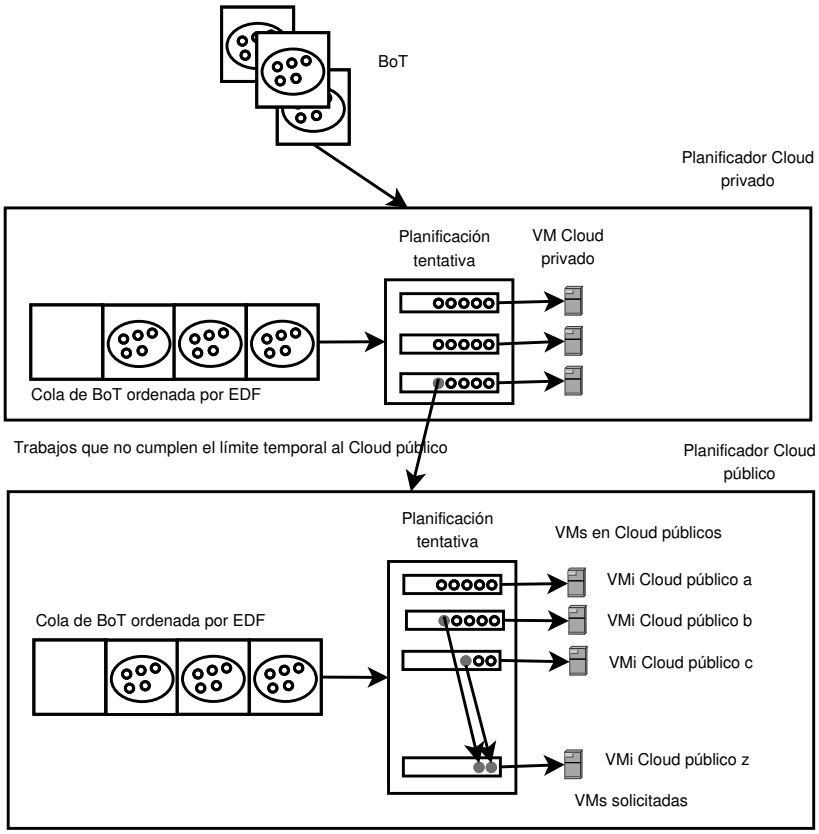



Figura 4.1: Arquitectura general del planificador

$$ec_{uajctik} = est_{uajctik} + ep_{uajctik} \tag{4.4}$$

El tiempo estimado de inicio de ejecución depende del tiempo estimado de finalización del trabajo que se ejecute con anterioridad en el procesador asignado para el trabajo. El planificador trabaja con tiempos de inicio y tiempos de procesamiento estimados porque es un planificador online y no es posible conocer con exactitud dichos tiempos de antemano. Para poder obtener los tiempos de procesamiento estimados, se introduce el concepto de estimador del tiempo de procesamiento.

**Definición 10** *El tiempo de procesamiento de un trabajo $j$ que forma parte de la aplicación $a$ del usuario $u$ en un procesador $k$ de la instancia de máquina virtual $i$ del tipo de máquina virtual $t$ del proveedor de Cloud $c$, es una variable aleatoria, continua y observable, $P_{uajctik}$, tomando valores en $\mathbb{R}^+$ y siguiendo una distribución de probabilidad desconocida con media y varianza finitas pero también desconocidas.*

**Definición 11** *Un estimador del tiempo de procesamiento, $E$, es una función que se emplea para producir una estimación, $ep_{uajctik}$, de los valores desconocidos de la variable aleatoria continua y observable $P_{uajctik}$. Nótese que $E$ es también una variable aleatoria.*

El planificador del Cloud privado depende de las estimaciones de los tiempos de procesamiento de los trabajos para identificar aquellos trabajos que no van a poder ser ejecutados antes de su límite temporal en el Cloud privado. Del mismo modo, el planificador del Cloud público depende de las estimaciones para calcular el tipo de máquina virtual más barato para ejecutar los trabajos en el Cloud público. Debido a que las estimaciones son imprecisas por definición, ambos planificadores producen planificaciones tentativas que pueden contener errores al estar basadas en información errónea. Los planificadores dependen de cuatro eventos para revisar las planificaciones tentativas e identificar y corregir errores produciendo nuevas planificaciones en base a la nueva información disponible:

1. Una nueva aplicación es enviada para su ejecución.
2. Una nueva instancia de máquina virtual está disponible en el Cloud público como resultado de una solicitud previa de aprovisionamiento.
3. Vence el tiempo estimado de inicio o finalización de la ejecución de un trabajo en la planificación tentativa actual.

4. Finaliza la ejecución de un trabajo.

Este enfoque basado en eventos simplifica otras estrategias empleadas en el estado del arte como el uso de intervalos periódicos de revisión de las planificaciones. El problema de las revisiones periódicas de las planificaciones es el ajuste del periodo temporal de revisión, tal y como se muestra en el trabajo de Van den Bossche et al [15].

Otro aspecto importante del planificador propuesto es que las aplicaciones enviadas para su ejecución son añadidas a una cola de aplicaciones de acuerdo a una política EDF (*Earliest Deadline First*). Esta política prioriza las aplicaciones que se encuentran más próximas en el tiempo a su límite temporal de ejecución y se sabe que es óptima en términos de límites temporales cumplidos en entornos con un solo procesador y no óptima en entornos multiprocesador [50]. Dado que bajo las restricciones del problema objeto de estudio no existe algoritmo de planificación óptimo [49], se escoge la política EDF ya que considera trabajos con límite temporal de ejecución y tiene un buen comportamiento general en sistemas con sobrecarga controlada [49].

## 4.1. Planificación en el Cloud privado

El planificador del Cloud privado se encarga de la planificación de las aplicaciones en el Cloud privado. Si una aplicación no puede ser ejecutada en el Cloud privado antes de su límite temporal es enviada al planificador del Cloud público. Para simplificar el algoritmo se asume que el Cloud privado tienen un número fijo de instancias de máquinas virtuales que no requieren de aprovisionamiento dinámico. En cualquier caso, el algoritmo puede ser extendido para aquellos casos en los que las instancias de máquinas virtuales en el Cloud privado son gestionadas de manera dinámica.

El algoritmo 1 muestra el pseudocódigo del algoritmo que procesa los eventos en el Cloud privado. Se trata de un algoritmo basado en eventos que ejecuta sus acciones cuando una nueva aplicación llega al sistema para su ejecución, cuando se cumplen el tiempo de finalización de una tarea en la planificación tentativa o cuando una tarea finaliza su ejecución. El algoritmo es sin desalojo, por lo que los trabajos no son interrumpidos una vez que comienzan su ejecución.

Tal y como se muestra en el algoritmo 2 las nuevas aplicaciones enviadas para su ejecución son insertadas en un cola que se encuentra ordenada según la política EDF (las aplicaciones con el límite temporal de ejecución más próximo se colocarán en primer lugar dentro de la cola).

**Algoritmo 1:** Gestión de eventos en el Cloud privado

**Input:** evt (Evento)
**Input:** PrivateVMs (Lista de instancias de máquinas virtuales en el Cloud privado)
**Input:** E (Función de estimación)

$S \leftarrow \{\}$ `// planificación`
$JobQueue \leftarrow []$ `// cola de trabajos a planificar`
$Met \leftarrow []$ `// Trabajos ejecutados antes de su límite`
$NotMet \leftarrow []$ `// Trabajos ejecutados fuera del límite`
**if** *evt es llegada nueva aplicación (app)* **then**
    $JobQueue \leftarrow JobQueue + [jobs \in app]$
    ordenar trabajos en JobQueue según EDF
    `// Invocar algoritmo 2`
    $S \leftarrow PlanificadorPrivado(JobQueue, PrivateVMs, E)$
**end**
**if** *evt es trabajo uaj finalizado en procesador ctik* **then**
    $avt_{ctik} \leftarrow now$
    **if** $C_{uajctik} \leq d_{ua}$ **then** $Met \leftarrow Met + [trabajo_{uaj}]$
    **else** $NotMet \leftarrow NotMet + [trabajo_{uaj}]$
    `// Invocar algoritmo 2`
    $S \leftarrow PlanificadorPrivado(JobQueue, PrivateVMs, E)$
**end**
**if** *evt es trabajo uaj debería finalizar de acuerdo a* $S$ **then**
    `// Invocar algoritmo 2`
    $S \leftarrow PlanificadorPrivado(JobQueue, PrivateVMs, E)$
**end**

La estrategia para generar la planificación en el Cloud privado se muestra en el algoritmo 2. El planificador construye una planificación tentativa empleando las estimaciones de los tiempos de procesamiento proporcionadas por el estimador configurado, el estado de ocupación de los procesadores de las instancias de máquinas virtuales disponibles y las aplicaciones en la cola. Esta planificación tentativa es reconstruida cuando se produce alguno de los eventos previamente comentados. Esta planificación tentativa es aplicada mediante la ejecución de las tareas sobre los procesadores de las instancias de máquinas virtuales, recogiendo al mismo tiempo el tiempo de uso de los mismos. Si el planificador no es capaz de construir una planificación tentativa que verifique el límite de ejecución de una tarea, esta tarea y todas las que estén pendientes de ejecución en la misma aplicación son enviadas al planificador del Cloud público para su ejecución.

La estrategia para inicializar la planificación se muestra en el algoritmo 3.

La estrategia para encontrar los procesadores que son viables para un trabajo, i.e. que permitirían ejecutar el trabajo antes de su límite temporal, se muestra en el algoritmo 4. Para cada tarea que se quiere añadir a la planificación se recorren todo los procesadores disponibles en el sistema tratando de encontrar uno que pueda ejecutar la tarea antes de su límite temporal. Para saber si un procesador puede ejecutar la tarea en plazo se tiene en cuenta el tiempo de procesamiento estimado de la tarea y el tiempo en el que se espera que el procesador pueda comenzar a ejecutar dicha tarea de acuerdo a la planificación actual para dicho procesador. Si más de un procesador puede ejecutar la tarea en plazo se selecciona uno al azar (en la implementación propuesta el último de la lista).

La figura 4.2 muestra un ejemplo del proceso de construcción de la planificación tentativa para una tarea. Cuando la aplicación a la que pertenece la tarea 8 es extraída de la cola de aplicaciones, todas las tareas de dicha aplicación son introducidas en la planificación. El planificador trata de añadir la tarea 8 a la planificación disponible en ese momento y mostrada en la parte (a) de la figura. La planificación disponible se materializa en una asignación de tareas a los procesadores de todas las instancias de máquinas virtuales, en este caso cuatro procesadores de tres instancias diferentes. La asignación tiene en cuenta el tiempo de procesamiento estimado de la tarea y que es representado como la longitud de la tarea en el eje X. El planificador itera todos los procesadores de todas las máquinas virtuales disponibles y comprueba si la tarea puede ser ejecutada en el procesador teniendo en cuenta el tiempo de finalización de las tareas que ya están planificadas en el mismo y el tiempo de procesamiento estimado de la tarea. En el ejemplo mostrado en la parte (b) de la figura, la tarea 8 solo puede ser ejecutada en plazo en el procesador 1 de la instancia de máquina

**Algoritmo 2:** Función PlanificadorPrivado

**Input:** JobQueue (Trabajos)
**Input:** PrivateVMs (Lista de instancias de máquinas virtuales en el Cloud privado)
**Input:** E (Función de estimación)
**Output:** S (Planificación)

```
1  S ← {} // planificación a construir
2  J ← [] // lista de trabajos a enviar al Cloud público
3  P ← [] // Lista de procesadores en el Cloud privado
   // Inicializar planificación. Algoritmo 3
4  S, P = InicializarPlanificacion(PrivateVMs, E)
5  ordenar procesadores en P por tiempo de disponibilidad (avt_ctik) más
     cercano
   // Encontrar un procesador viable para cada trabajo
6  for job_uaj ∈ JobQueue do
       // Algoritmo 4
7      FeasibleProcessors ←
         EncontrarProcesadoresViables(u, a, j, P, E)
8      if FeasibleProcessors = ∅ then
9          J ← J + job_uaj
10     end
11     else
12         ctik ← procesador al frente de FeasibleProcessors
13         S ← S + (u, a, j, c, t, i, k, avt_ctik, avt_ctik + E(u, a, j, c, t, i, k))
14         avt_ctik ← avt_ctik + E(u, a, j, c, t, i, k)
15     end
16 end
17 ejecutar S en los procesadores disponibles de PrivateVMs
18 enviar los trabajos en J al planificador del Cloud público
19 return S
```

**Algoritmo 3:** Función InicializarPlanificacion

**Input:** VMs (Lista de instancias de máquinas virtuales en el Cloud)
**Input:** E (Función de estimación)
**Output:** S (Planificación)
**Output:** P (Lista de procesadores)

```
1  S ← {} // planificación a construir
2  P ← [] // Lista de procesadores en el Cloud
   // inicializar la planificación con los trabajos en
      ejecución
3  for instancia de máquina virtual cti en VMs do
4      for procesador k en la instancia cti do
5          if procesador k tiene un trabajo uaj en ejecución then
6              ep_uajctik ← E(u, a, j, c, t, i, k)
7              avt_ctik ← st_uajctik + ep_uajctik
8              S ← S + (u, a, j, c, t, i, k, st_uajctik, avt_ctik)
9          end
10         else
11             avt_ctik ← now
12         end
13         P ← P+ procesador ctik
14     end
15 end
16 return S,P
```

**Algoritmo 4:** Función EncontrarProcesadoresViables

**Input:** u,a,j (Índices de un trabajo)
**Input:** P (Lista de procesadores disponibles)
**Input:** E (Función de estimación)
**Output:** ProcesadoresViables (Lista de procesadores para ejecutar el trabajo dentro del límite temporal)

```
1 ProcesadoresViables ← [];
2 for procesador ctik en P do
3 |   if avt_ctik + E(u, a, j, c, t, i, k) < d_ua then
4 |   |   ProcesadoresViables ← ProcesadoresViables + procesador ctik;
5 |   end
6 end
7 return ProcesadoresViables
```

virtual 2.

## 4.2. Planificación en el Cloud público

El algoritmo de planificación en el Cloud público es un algoritmo basado en cuatro eventos. Para simplificar dicho algoritmo se considera que cada proveedor de Cloud público tiene una capacidad infinita y que es capaz de aprovisionar un número infinito de instancias de máquinas virtuales. Esta suposición no es verdadera en un entorno real pero el algoritmo puede ser extendido fácilmente para considerar un número máximo de instancias a aprovisionar en cada proveedor.

El algoritmo 5 muestra el pseudocódigo para la gestión de eventos en el Cloud público. El algoritmo es muy similar al presentado para el Cloud privado con la excepción de que en el Cloud público se añade un nuevo evento que dispara la regeneración de la planificación tentativa: la disponibilidad de una nueva instancia de máquina virtual.

La planificación tentativa es reconstruida cuando una nueva instancia de máquina virtual está disponible, pudiendo así corregir el error en la planificación causado por el retraso en el tiempo de aprovisionamiento de nuevas instancias. Tal y como muestran Mao et al [51] el tiempo de aprovisionamiento y arranque de las instancias de máquinas virtuales varía dependiendo del momento del día y del tipo de máquina virtual.

El algoritmo 6 muestra el pseudocódigo para construir y ejecutar la planifi-

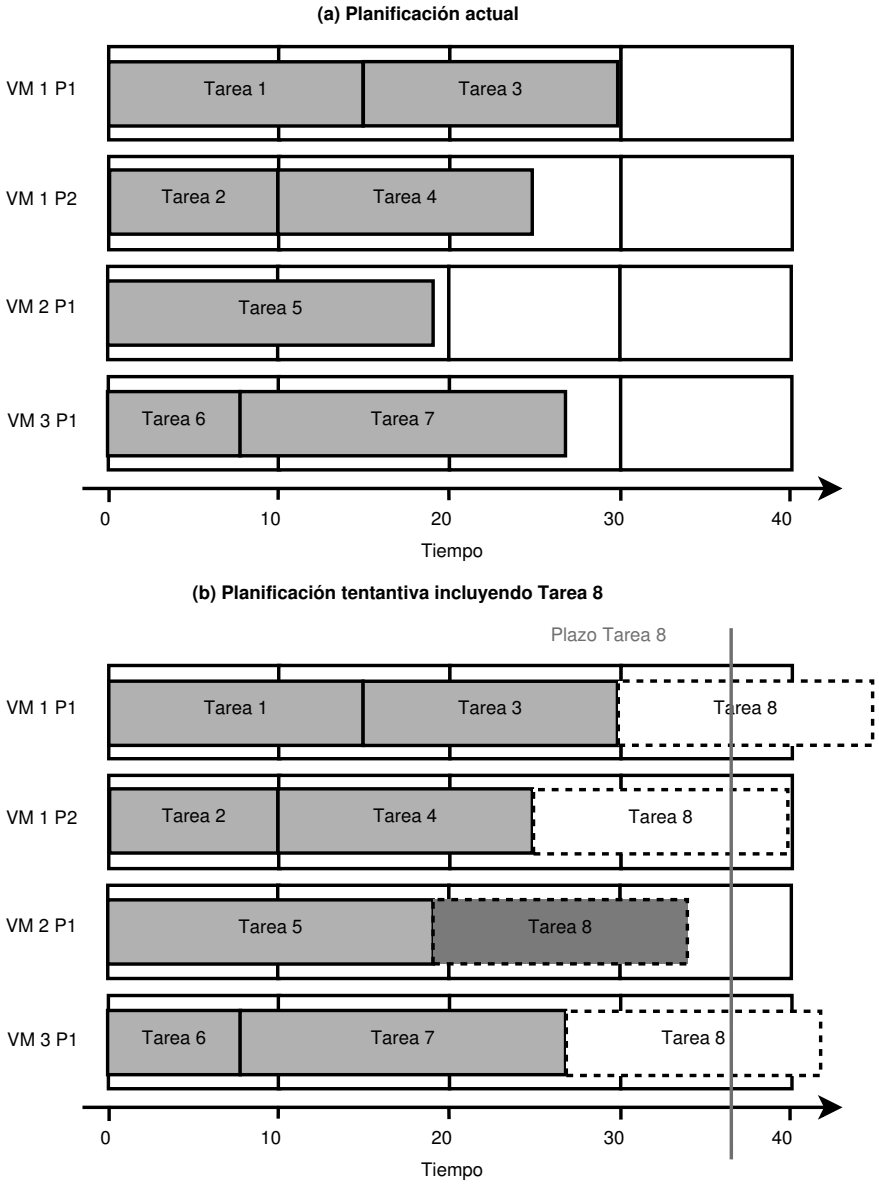



Figura 4.2: Ejemplo de planificación tentativa

**Algoritmo 5:** Gestión de eventos en el Cloud público

**Input:** evt (Event)
**Input:** VMTypes (Lista de tipos de máquinas virtuales disponibles)
**Input:** E (Función de estimación)
**Output:** C (Coste total)

$S \leftarrow \{\}$ `// planificación`
$C \leftarrow 0$ `// Coste total, a 0 cuando se arranca el planificador`
$PublicJobQueue \leftarrow []$ `// cola de trabajos a planificar`
$VMs \leftarrow []$ `// Lista de instancias de máquinas virtuales`
$Met \leftarrow []$ `// Trabajos ejecutados antes de su límite`
$NotMet \leftarrow []$ `// Trabajos ejecutados fuera del límite`
**if** *evt es llegada nuevos trabajos* **then**
    $PublicJobQueue \leftarrow PublicJobQueue + trabajos$
    ordenar trabajos en PublicJobQueue según EDF
    `// Algoritmo 6`
    $S \leftarrow PlanificadorPublico(PublicJobQueue, VMs, VMTypes, E)$
**end**
**if** *evt es trabajo uaj finalizado en procesador ctik* **then**
    $avt_{ctik} \leftarrow now$
    **if** $C_{uajctik} \leq d_{ua}$ **then** $Met \leftarrow Met + [trabajo_{uaj}]$
    **else** $NotMet \leftarrow NotMet + [trabajo_{uaj}]$
    **if** *instancia de máquina virtual cti no está en S* **then**
        Apagar instancia de máquina virtual cti
        $sdt_{cti} \leftarrow now$
        `// Añadir coste de máquina virtual cti`
        $C \leftarrow C + Ceilling(sdt_{cti} - avt_{cti}, bp_c) \cdot cvmt_{ct}$
    **end**
    $S \leftarrow PlanificadorPublico(PublicJobQueue, VMs, VMTypes, E)$
**end**
**if** *evt es nueva instancia de máquina virtual cti disponible* **then**
    $VMs \leftarrow VMs +$ instancia cti
    **for** *procesador ctik en la instancia cti* **do**
        $avt_{ctik} \leftarrow now$
    **end**
**end**
**if** *evt es trabajo uaj debería finalizar de acuerdo a S* **then**
    $S \leftarrow PlanificadorPublico(PublicJobQueue, VMs, VMTypes, E)$
**end**

cación en el Cloud público. El planificador construye la planificación tentativa iterando la lista de aplicaciones ordenada según la política EDF y buscando para cada tarea un procesador que permita ejecutarla en plazo. Esta búsqueda de un procesador viable para ejecutar la tarea en plazo se realiza de la misma forma que en el Cloud privado.

En el caso de que no exista ningún procesador capaz de ejecutar una tarea en plazo, el algoritmo busca un tipo de máquina virtual que sea capaz de ejecutarla en plazo según la estimación del tiempo de procesamiento de la tarea en dicho tipo de máquina virtual (ver algoritmo 7). Si más de un tipo de máquina virtual es capaz de ejecutar la tarea en plazo se selecciona el tipo más barato. En el caso de encontrar un tipo de máquina virtual viable, el planificador solicita la creación de una nueva instancia de dicho tipo y marca la tarea para ser ejecutada en dicha instancia cuando ésta se encuentre disponible. Como se muestra en el bucle de la línea 16, dado que el algoritmo no tiene en cuenta el tiempo de aprovisionamiento de las instancias de máquina virtual, los procesadores de la instancia solicitada se consideran de manera inmediata en la planificación, aunque la ejecución de las tareas en dichos procesadores comenzará solo cuando la instancia este realmente lista.

## 4.3. Estimadores del tiempo de procesamiento

Este trabajo propone cuatro estimadores de los tiempos de procesamiento de los trabajos para poder estudiar el impacto de dichas estimaciones sobre el planificador: estimador perfecto, estimador de usuario, estimador de media y estimador de Chebyshev.

### 4.3.1. Estimador perfecto

El estimador perfecto, $EP$, es un estimador irreal que proporciona el tiempo de procesamiento exacto de un trabajo en una instancia de máquina virtual. Se incluye solo con fines de validación y su implementación no sería posible en un sistema real. Su definición como función se realiza de acuerdo a la ecuación 4.5.

$$EP(u, a, j, c, t, i, k) = p_{uajctik} \tag{4.5}$$

Es posible calcular el tiempo de procesamiento exacto de la tarea $j$ perteneciente a la aplicación $a$ del usuario $u$ en el procesador $k$ del tipo de máquina virtual $t$ del proveedor de Cloud público $c$ mediante la ecuación 4.6.

**Algoritmo 6:** Función PlanificadorPublico

**Input:** PublicJobQueue (Trabajos a ejecutar en el Cloud público)
**Input:** VMInstances (Lista de instancias de VMs en el Cloud público)
**Input:** VMTypes (Lista de tipos de VMs en el Cloud público)
**Input:** E (Función de estimación)
**Output:** S (Planificación)

```
1  S ← {} // planificación a construir
2  U ← [] // lista de trabajos inviables
3  P ← [] // Lista de procesadores en el Cloud privado
   // Inicializar planificación. Algoritmo 3
4  S, P = InicializarPlanificacion(VMInstances, E)
5  ordenar P por tiempo de disponibilidad (avt_ctik) más cercano
6  for job_uaj ∈ PublicJobQueue do
       // Algoritmo 4
7      FeasibleProcessors ←
        EncontrarProcesadoresViables(u, a, j, P, E)
       // Sin procesadores viables, solicitar nueva VM
8      if FeasibleProcessors = ∅ then
           // Algoritmo 7, buscar tipo de VM viable
9          FeasibleTypes ← BuscarTiposValidos(u, a, j, VMTypes, E)
10         if FeasibleTypes = ∅ then
11             U ← U+ trabajo uaj
12         end
13         else
               // Algoritmo 8, solicitar nueva instancia y
                  actualizar planificación
14             S ← SolicitarNuevaInstancia(S, u, a, j, FeasibleTypes, E)
15         end
16     end
17     else
18         ctik ← procesador con minimo cvmt_ct en FeasibleProcessors
19         S ← S + (u, a, j, c, t, i, k, avt_ctik, avt_ctik + E(u, a, j, c, t, i, k))
20         avt_ctik ← avt_ctik + E(u, a, j, c, t, i, k)
21     end
22 end
23 ejecutar S en los procesadores disponibles de VMInstances
24 return S
```

**Algoritmo 7:** Función BuscarTiposValidos

**Input:** u,a,j (Índices de un trabajo)
**Input:** VMTypes (Lista de tipos de máquinas virtuales en todos los proveedores de Cloud público)
**Input:** E (Función de estimación)
**Output:** FeasibleTypes (Tipos de instancias de máquinas virtuales válidos)

```
1  FeasibleTypes ← []
2  for type_ct ∈ VMTypes do
3  |  if now + E(u, a, j, c, t) < d_ua then
4  |  |  FeasibleTypes ← FeasibleTypes + tipo ct
5  |  end
6  end
7  return FeasibleTypes
```

**Algoritmo 8:** Función SolicitarNuevaInstancia

**Input:** S (Planificación a modificar)
**Input:** u,a,j (Índices del trabajo para el que se solicita nueva instancia)
**Input:** FeasibleTypes (Lista de tipos de VMs)
**Input:** E (Función de estimación)
**Output:** S (Planificación modificada)

```
1  SelectedType ← tipo ct con mínimo cvmt_ct de FeasibleTypes
2  pedir nueva instancia cti de tipo SelectedType
   // Añadir los procesadores de cti a la planificación
3  for procesador k de la instancia cti do
   |  // asignar trabajo a primer procesador
4  |  if k = 0 then
5  |  |  ep_uajctik ← E(u, a, j, c, t, i, k)
6  |  |  avt_ctik ← st_uajctik + ep_uajctik
7  |  |  S ← S + (u, a, j, c, t, i, k, st_uajctik, avt_ctik)
8  |  end
9  |  else  avt_ctik ← now // procesador libre
10 |
11 end
12 return S
```

$$p_{uajctik} = \frac{l_{uaj}}{v_{ctk}} \tag{4.6}$$

Combinando las ecuaciones 4.5 y 4.6 se puede definir el estimador perfecto como la función expresada en la ecuación 4.7.

$$EP(u, a, j, c, t, i, k) = \frac{l_{uaj}}{v_{ctk}} \tag{4.7}$$

Nótese que en un entorno real los valores $l_{uaj}$ y $v_{ctk}$ no son conocidos y el estimador perfecto solo puede ser implementado en el ámbito del simulador que se empleará en este trabajo durante la fase de validación. El resto de estimadores y el propio algoritmo de planificación no tienen acceso a estos valores tal y como ocurriría en un sistema real.

### 4.3.2. Estimador de usuario

El estimador de usuario, *EU*, trata de imitar el esfuerzo del usuario final por proporcionar una estimación de los tiempos de procesamiento. Este estimador se basa en el principio de que un mismo usuario envía principalmente aplicaciones similares para su ejecución. La precisión de este estimador viene dada por cómo el usuario escoge el valor de la estimación y las características de la distribución de probabilidades de los tiempos de procesamiento de las tareas de dicho usuario. En este trabajo se emplea un enfoque común en otros trabajos y el estimador retorna un valor constante ($ce_u$) para todos los trabajos de un mismo usuario, independientemente del tipo de máquina virtual. El estimador de usuario se puede expresar como función de acuerdo a la ecuación 4.8.

$$EU(u, a, j, c, t, i, k) = ce_u \tag{4.8}$$

Durante las simulaciones llevadas a cabo en este trabajo el valor de la constante $ce_u$ se calcula a partir del tiempo base de procesamiento de cada usuario $u$ y usando este valor como la media de un generador de números aleatorios que sigue una distribución normal con una desviación igual a 0,5 veces la media. El tiempo base de procesamiento de un usuario es un parámetro empleado para la generación de las cargas de trabajo de un usuario y se refiere a la media de los tiempos de procesamiento dentro de la distribución de probabilidad de dichos tiempos. En un entorno real el valor de la estimación sería obtenido mediante técnicas de perfilado (del inglés *profiling*) por parte del usuario.

### 4.3.3. Estimadores de media y de Chebyshev

Teniendo en cuenta la definición del tiempo de procesamiento como una variable aleatoria continua (definición 10), sería posible emplear los parámetros de la función de distribución de dicha variable como estimadores de los tiempos de procesamiento. Así, resultaría lógico aproximar el tiempo de procesamiento por el valor de la esperanza de la función de distribución, dado que este es un parámetro de centralización o localización de los valores de la variable aleatoria. El problema es que la función de distribución de los tiempos de procesamiento es totalmente desconocida por el sistema, tanto en su forma (la familia de distribuciones), como en los parámetros concretos de la misma.

Aunque la función de distribución de los tiempos de procesamiento sea desconocida, el sistema sí que puede obtener observaciones de dicha variable simplemente almacenando el histórico de tiempos de procesamiento de las tareas que se ejecutan. Esto permite al sistema obtener una muestra de la variable aleatoria y emplear posteriormente la inferencia estadística para obtener una estimación puntual de los parámetros de la distribución poblacional a partir de estadísticos de la muestra. Por el teorema de Glivenko-Cantelli (asociado a la ley fuerte de los grandes números) [52] se sabe que cuando el tamaño de la muestra aumenta, la función de distribución empírica derivada de la muestra converge de manera *casi segura* a la verdadera función de distribución de la población. Empleando el método de estimación puntual de los momentos de Pearson [53] se tendría entonces que la esperanza poblacional puede ser estimada por la media muestral. Este estimador sería consistente, aunque no centrado ni eficiente, pero al no conocer la familia de la función de distribución de la variable aleatoria no es posible realizar la estimación mediante el método máxima verosimilitud, que sí permite obtener estimadores de mejor calidad.

El estimador de media se basa en este concepto y genera estimaciones de los tiempos de procesamiento calculando la media muestral de los tiempos de procesamiento de cada usuario en cada tipo de máquina virtual, dado que se asume que seguirán un mismo patrón. Como mostraron Feitelson y Nitzberg [54] los usuarios tienden a enviar normalmente el mismo tipo de aplicaciones y trabajos a ejecución. Los autores muestran cómo para el sistema en estudio el coeficiente de variación de los tiempos de procesamiento de los trabajos es en la mayoría de los casos inferior a uno. Los autores indican que este resultado invita al diseño de estimadores de los tiempos de procesamiento que sean eficientes. Posteriormente, Gibbons [55] demostró, tras analizar las trazas de tres sistemas de supercomputación, que categorizando los trabajos de acuerdo al nombre del ejecutable, el grado de paralelismo y el nombre del usuario que los ejecuta, se

obtienen coeficientes de variación en los tiempos de procesamiento inferiores a los coeficientes de variación del sistema en global. Si se asume que los tiempos de procesamiento de las tareas de un mismo usuario serán similares en instancias de máquinas virtuales del mismo tipo e independientes del procesador dentro de la instancia de máquina virtual se puede simplificar la definición 10 para prescindir de la información relativa al trabajo concreto, $j$, la aplicación, $a$, la instancia de máquina virtual, $i$, y el procesador, $k$. Esto permite introducir $P_{uct}$ de acuerdo a la definición 12.

**Definición 12** *El tiempo de procesamiento de un trabajo del usuario u en una instancia de máquina virtual del tipo t del proveedor de Cloud c es una variable aleatoria continua y observable, $P_{uct}$, tomando valores en $\mathbb{R}^+$ y siguiendo una distribución de probabilidad desconocida con media y varianza finitas pero también desconocidas.*

Durante la ejecución del planificador el sistema es capaz de recoger $n$ observaciones de la variable $P_{uct}$ mediante la medida de los tiempos de procesamiento de las tareas que finalizan su ejecución, obteniendo de esta manera un muestra aleatoria de tamaño $n$ de la variable aleatoria $P_{uct}$. Esta muestra se define de acuerdo a la ecuación 4.9. Los valores de la muestra pueden ser posteriormente empleados para calcular la media muestral, $m_{uct}$, y la varianza muestral, $s_{uct}$, de la variable $P_{uct}$ de acuerdo a las ecuaciones 4.10 y 4.11 respectivamente.

$$p_{uct} = \{p_{uct1}, ..., p_{uctn}\} \tag{4.9}$$

$$m_{uct} = \frac{1}{n}\sum_{i=1}^{n} p_{ucti} \tag{4.10}$$

$$s_{uct} = \frac{1}{n-1}\sum_{i=1}^{n}(p_{ucti} - m_{uct})^2 \tag{4.11}$$

El estimador de media, $EM$, es un estimador basado en muestras que proporciona estimaciones en base a la media móvil muestral de las últimas $n$ medidas de los tiempos de procesamiento. Este estimador esta basado en los resultados del trabajo de Tsafrir et al [39]. Como función, el estimador de media se define de acuerdo a la ecuación 4.12. En este trabajo $n$ se fija a un valor constante de 100 para que coincida con el valor empleado en el estimador de Chebyshev, dado que permite garantizar ciertas propiedades de ese estimador como se verá más adelante. El valor de $n$ determina si usar datos recientes es más importante que

el histórico completo de trabajos similares y por tanto la capacidad del planificador para adaptarse a variaciones en la carga de entrada y en la infraestructura subyacente. La desventaja de este estimador es que la precisión de la media como estimador depende de la forma de la distribución de probabilidad de los tiempos de procesamiento. La media aritmética se ve muy afectada por valores extremos y no es un valor apropiado para distribuciones altamente sesgadas. Cuando el estimador no cuenta con medidas suficientes para producir su salida para un usuario y tipo de máquina virtual se emplea el valor del estimador de usuario.

$$EM(u, a, j, c, t, i, k) = m_{uct} \tag{4.12}$$

Desde el punto de vista del algoritmo de planificación y el objetivo primario de ejecutar las tareas antes de su límite temporal, es importante que las estimaciones de los tiempos de procesamiento sean siempre mayores o iguales que el tiempo real de procesamiento (ecuación 4.13) para poder maximizar el número de tareas ejecutadas en plazo (minimizar la tardanza total de la planificación).

$$p_{uajctik} \leq ep_{uajctik} \tag{4.13}$$

Durante la construcción de las planificaciones tentativas el planificador tiene que garantizar que el tiempo esperado de finalización de cada tarea sea inferior al límite temporal de la misma de acuerdo a la ecuación 4.14.

$$ec_{uajctik} < d_{ua} \tag{4.14}$$

Teniendo en cuenta la ecuación 4.4, la condición de la ecuación 4.14 se puede transformar en la condición de la ecuación 4.15. Durante la construcción de las planificaciones tentativas el planificador siempre garantiza que dicha condición se cumple o en caso contrario la ejecución de la tarea será abortada y se marcará como inviable.

$$est_{uajctik} + ep_{uajctik} < d_{ua} \tag{4.15}$$

Si $p_{uajctik} \leq ep_{uajctik}$ y $est_{uajctik} + ep_{uajctik} < d_{ua}$ entonces se garantiza que $est_{uajctik} + p_{uajctik} < d_{ua}$, independientemente del valor de $p_{uajctik}$. En otras palabras, si las estimaciones de los tiempos de procesamiento son siempre mayores o iguales que los tiempos de procesamiento reales, el planificador podrá garantizar que la planificación generada permite ejecutar las tareas en plazo. El valor de $est_{uajctik}$ no es relevante en este análisis ya que es calculado por el planificador a partir de otros valores.

Aunque las estimaciones de los tiempos de procesamiento mayores o iguales que los tiempos de procesamiento reales ayudan al planificador a minimizar la tardanza total de la planificación (objetivo primario de la planificación), pueden también incrementar el coste total de la planificación (objetivo secundario). Con estimaciones mayores de los tiempos de procesamiento el planificador tenderá a contratar un mayor número de máquinas en el Cloud público, siendo dichas máquinas más potentes y por tanto más caras. Estimaciones excesivamente altas producirán también con mucha probabilidad el aborto de las tareas y su clasificación como inviables por parte del planificador.

Debe notarse que el estimador de media $EM$ no garantiza que todas las estimaciones producidas sean iguales o mayores que el tiempo de procesamiento real. Esto se debe a que dicho estimador se basa en emplear la media muestral como estimador de la esperanza de la distribución de los tiempos de procesamiento. Al ser una estimación puntual de un parámetro de la función de distribución poblacional no ofrece ningún nivel de confianza de la estimación y por tanto no podemos saber su relación con los valores reales de la variable aleatoria.

El objetivo sería poder definir un estimador que permitiese generar estimaciones que fuesen mayores que los valores de procesamiento reales con una determinada probabilidad:

$$P(P_{uajctik} \leq ep_{uajctik}) = 1 - \alpha \tag{4.16}$$

O lo que es lo mismo, estimaciones con una determinada probabilidad de ser menores, es decir inválidas desde el punto de vista del planificador:

$$P(P_{uajctik} > ep_{uajctik}) = \alpha \tag{4.17}$$

Si se conociese la función de densidad de probabilidad $f_{uajctik}(t)$ de la variable aleatoria $P_{uajctik}$ que representa el tiempo de procesamiento, obtener el valor de $ep_{uajctik}$ para un $\alpha$ dado, (al que denominaremos $ep_{uajctik\alpha}$) sería sencillo empleando la propia definición de función de densidad de una variable aleatoria:

$$P(P_{uajctik} > ep_{uajctik\alpha}) = \int_{ep_{uajctik\alpha}}^{\infty} f_{uajctik}(t)dt = \alpha \tag{4.18}$$

Para determinar el valor $ep_{uajctik\alpha}$ que satisface la ecuación 4.18 basta con obtener la función de distribución de probabilidad de $P_{uajctik}$ denotada por $F_{uajctik}(ep)$ tal y como se muestra en la ecuación 4.19 y obtener a continuación el valor de $ep_{uajctik\alpha}$ que verifica la ecuación 4.20.

$$F_{uajctik}(ep) = \int_{-\infty}^{ep} f_{uajctik}(t)dt \tag{4.19}$$

$$F_{uajctik}(ep_{uajctik\alpha}) = 1 - \alpha \tag{4.20}$$

Dado que el sistema no conoce la función de densidad de probabilidad de los tiempos de procesamiento, no es posible emplear la ecuación 4.20 para obtener una estimación que garantice, con una determinada probabilidad, que es mayor que los tiempos de procesamiento reales. Aunque no sea posible obtener el valor $ep_{uajctik\alpha}$ debe notarse que cualquier valor $ep_{uajctik}$ obtenido de manera que $ep_{uajctik} \geq ep_{uajctik\alpha}$ será válido desde el punto de vista del planificador dado que permitirá generar planificaciones válidas respetando la condición de la ecuación 4.15 con una probabilidad menor o igual que $\alpha$. De manera matemática tenemos que si $ep_{uajctik} \geq ep_{uajctik\alpha}$ entonces se verifica la ecuación 4.21.

$$P(P_{uajctik} > ep_{uajctik}) = \int_{ep_{uajctik}}^{\infty} f_{uajctik}(t)dt \leq \alpha \tag{4.21}$$

El valor $ep_{uajctik}$ acota la probabilidad de la variable aleatoria $P_{uajctik}$ entorno a $\alpha$. Aunque no se conozca la distribución de probabilidad de $P_{uajctik}$, el teorema de Chebyshev permite acotar la probabilidad de una variable aleatoria alrededor de su esperanza y nos permitirá encontrar el valor de la cota $ep_{uajctik}$.

**Teorema 13 (Desigualdad de Chebyshev)** *Para una variable aleatoria $X$ de esperanza $E(X)$ y varianza $Var(X)$ finitas se verifica que para todo $k > 0$:*

$$P(|X - E(X)| \geq k \cdot \sqrt{Var(X)}) \leq \frac{1}{k^2} \tag{4.22}$$

*De manera alternativa se puede reformular la ecuación como:*

$$P(E(X) - k \cdot \sqrt{Var(X)} \leq X \leq E(X) + k \cdot \sqrt{Var(X)}) \geq 1 - \frac{1}{k^2} \tag{4.23}$$

La desigualdad de Chebyshev garantiza que en cualquier distribución de probabilidad no más de $1/k^2$ de los valores de la distribución pueden estar alejados de la esperanza $k$ veces la desviación típica. Dicho de otro modo, al menos $1 - 1/k^2$ de los valores de la distribución se encuentran dentro del intervalo de confianza definido por la esperanza y $k$ veces la desviación típica. La desigualdad se puede emplear para obtener cotas de variables aleatorias siguiendo distribuciones de probabilidad desconocidas. Empleando valores de $k$ concretos tenemos

que la desigualdad garantiza que un mínimo del 75 % de los valores caen dentro del intervalo definido por dos veces la desviación típica y la media y un 89 % de los valores caen dentro del intervalo definido por tres veces la desviación típica.

Si $k$ toma el valor $\alpha^{-1/2}$, considerando $\alpha$ como la probabilidad empleada en las ecuaciones 4.16 y 4.17 se puede reescribir la ecuación del teorema de Chebyshev 4.23 como sigue:

$$P(E(X) - \alpha^{-1/2} \cdot \sqrt{Var(X)} \leq X \leq E(X) + \alpha^{-1/2} \cdot \sqrt{Var(X)}) \geq 1 - \alpha \quad (4.24)$$

De la ecuación 4.24 y las propiedades de las funciones de densidad de probabilidad se tiene que la desigualdad de la ecuación 4.25 también debe ser cierta:

$$P(X \leq E(X) + \alpha^{-1/2} \cdot \sqrt{Var(X)}) \geq 1 - \alpha \quad (4.25)$$

Siendo la expresión de la ecuación 4.25 similar a la de la ecuación 4.16 podemos por tanto afirmar que es posible emplear el valor $E(X) + \alpha^{-1/2} \cdot \sqrt{Var(X)}$ como estimación de los tiempos de procesamiento. La ecuación 4.26 muestra la probabilidad complementaria a la mostrada en la ecuación 4.25.

$$P(X > E(X) + \alpha^{-1/2} \cdot \sqrt{Var(X)}) \leq \alpha \quad (4.26)$$

A partir de la ecuación 4.26 se puede garantizar que la probabilidad de que el valor real sea mayor que la estimación así generada es menor o igual que $\alpha$. En la ecuación 4.17 (complementaria de la ecuación 4.16) la cota buscada tenía una probabilidad igual a $\alpha$ de que el valor real fuese mayor, mientras que la cota obtenida mediante la desigualdad de Chebyshev tiene una probabilidad menor o igual a $\alpha$, de ahí que la cota pueda ser superior a la cota mínima posible, siendo esta la causa de que estas estimaciones se consideren pesimistas dado que son iguales o superiores a la cota mínima buscada.

Dado que la distribución de los tiempos de procesamiento es desconocida, también su esperanza y varianza poblacional son desconocidas y por tanto no es posible realizar el cálculo exacto de la cota $E(X) + \alpha^{-1/2} \cdot \sqrt{Var(X)}$. Aunque al igual que en el caso del estimador de media sería posible estimar la esperanza poblacional mediante la media muestral y la varianza poblacional mediante la varianza muestral, estas aproximaciones no garantizarían la probabilidad $\alpha$ deseada.

Diversos autores han extendido la desigualdad de Chebyshev para aquellos casos en los que la esperanza y la varianza poblacional no son conocidos, pero donde la media y la varianza muestral derivada de $r$ muestras de la distribución

están disponibles. Kabán [56] propone la siguiente desigualdad siendo $r \geq 2$, $m$ la media muestral y $s$ la desviación típica muestral para una muestra de tamaño $r$ y $\epsilon > 0$:

$$P(|X - m| \geq \epsilon m) \leq \frac{r^2 - 1}{r^2} \frac{1}{\epsilon^2} \frac{s^2}{m^2} + \frac{1}{r} \tag{4.27}$$

Si $\epsilon = \frac{ks}{m}$ entonces la ecuación 4.27 se puede reescribir según la ecuación 4.28:

$$P(|X - m| \geq ks) \leq \frac{r^2 - 1}{r^2} \frac{1}{k^2} + \frac{1}{r} \tag{4.28}$$

La ecuación 4.28 se puede considerar la versión para datos muestrales de la ecuación original del teorema de Chebyshev mostrada en 4.22 y que trabajaba con la esperanza y la desviación típica poblacional. La diferencia en la probabilidad calculada por ambas ecuaciones se muestra en la figura 4.3. Mientras que la ecuación de Chebyshev emplea la esperanza y desviación típica poblacional y por tanto no se ve afectada por el número de muestras, la ecuación de Kabán emplea la media y desviación típica muestrales y por tanto el valor de la probabilidad varia con el tamaño de la muestra.

Cabe notar que a partir de un tamaño de muestra de 100 ($r \geq 100$) la diferencia entre las probabilidades de ambas ecuaciones es menor de 0,01. Esto implica que a efectos prácticos y cuando el tamaño de muestra es lo suficientemente grande, se puede emplear la ecuación de Chebyshev para calcular el valor de $k$ a partir de una probabilidad ($\alpha$) deseada, aunque se estén empleando la media y desviación típica muestral en lugar de los valores poblacionales. Dada una probabilidad de error deseada ($\alpha$), el valor de $k$ que permite obtener la cota de los tiempos de procesamiento de acuerdo a la desigualdad de Chebyshev (ecuación 4.25) se calculará de acuerdo a la ecuación 4.29.

$$k = \alpha^{-1/2} \tag{4.29}$$

Si se desea que $\alpha$ tome el valor 0,25 (la probabilidad de que los valores reales de la distribución sean mayores que la cota es igual o inferior a 0,25) el valor de $k$ a emplear es 2. Empleando un tamaño de muestra de 100 ($r = 100$) y con $k = 2$, la ecuación de Kabán arroja una probabilidad del 0,259975 cuando se emplea la media y desviación típica muestral.

En base a este desarrollo matemático se propone el estimador de Chebyshev, $EC$, un estimador basado en medidas del tiempo de procesamiento, la desigualdad de Chebyshev [57] y el concepto del tiempo de procesamiento en el peor

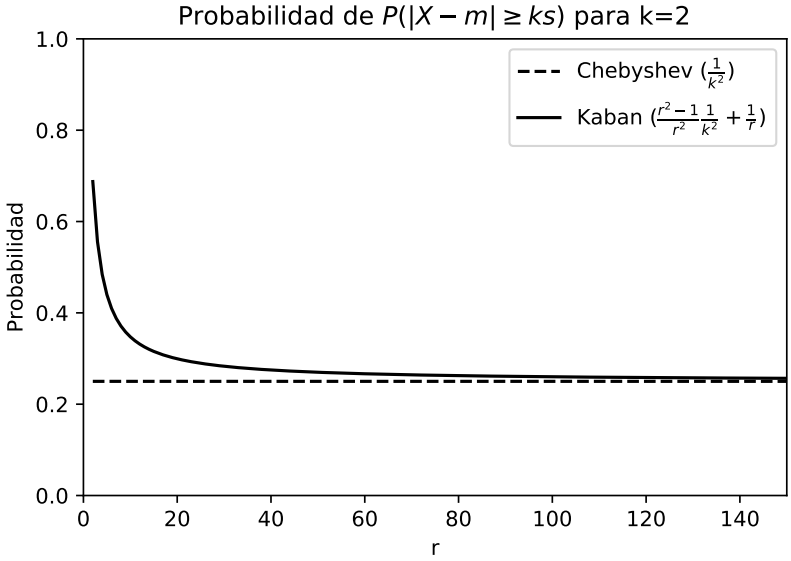



Figura 4.3: Probabilidad de que los valores de una distribución sean mayores que la cota de Chebyshev y la cota de Kabán.

caso (en inglés *worst case execution time - WCET*) [58]. De acuerdo al desarrollo realizado, la desigualdad de Chebyshev permite obtener cotas superiores del tiempo de procesamiento de las tareas de un usuario en un tipo de máquina virtual. Esta cota superior proporcionada por el estimador se aproxima de forma práctica a la definición del tiempo de procesamiento en el peor caso (WCET), una estimación pesimista comúnmente empleada en el análisis de sistemas en tiempo real.

En este trabajo se fija un valor de $k$ igual a 2 en la desigualdad (la probabilidad de que el valor del tiempo de procesamiento sea mayor que la estimación es menor o igual a 0,25) y se emplea la media y desviación estándar muestral para expresar el estimador de Chebyshev según la ecuación 4.30. La media ($m_{uct}$) y desviación típica muestral ($s_{uct}$) se calculan sobre las últimas 100 muestras para minimizar el error que se produce al emplear los estadísticos muestrales y no los parámetros de la distribución, que es desconocida. Cuando no existen muestras suficientes el estimador retorna el mismo valor que el estimador de usuario.

$$EC(u, a, j, c, t, i, k) = m_{uct} + 2s_{uct} \tag{4.30}$$

# 5

# Evaluación

Este capítulo presenta la evaluación de la estrategia de planificación propuesta con diferentes cargas de trabajo y escenarios y su comparación con otro algoritmo de planificación del estado del arte.

## 5.1. Metodología

Esta sección describe el proceso de evaluación llevado a cabo, las herramientas empleadas y los métodos de recogida de datos y análisis.

Las simulaciones se han centrado en evaluar el rendimiento de la estrategia con diferentes estimadores de los tiempos de procesamiento y diferentes cargas de trabajo, bajo dos escenarios de simulación que pretenden estudiar el impacto de variar el límite temporal de ejecución y el tiempo entre llegadas de las aplicaciones. Además, las simulaciones estudian también el impacto del tiempo de aprovisionamiento de las instancias de máquinas virtuales.

En un esfuerzo por comparar el planificador propuesto con otros planificadores en el estado del arte, se ha implementado también el algoritmo de planifica-

ción propuesto por Van den Bossche et al [15] de modo que las simulaciones se han ejecutado también con dicho planificador para las mismas cargas de trabajo y condiciones. El trabajo de Van den Bossche et al ha sido seleccionado porque el modelo de entorno Cloud y la carga de trabajo empleada para la validación en dicho trabajo es muy similar a la empleada en este trabajo. Debe tenerse en cuenta que aunque el modelo de simulación presentado en este trabajo no tiene en cuenta la latencia de la red de comunicaciones, los escenarios de simulación empleados son más exigentes en términos de límites temporales de ejecución y en cuanto al tamaño del Cloud privado que los del trabajo de Va den Bossche et al.

Combinando el planificador propuesto en este trabajo, el planificador de referencia del estado del arte y los cuatro estimadores de los tiempos de procesamiento se tiene que cada escenario de simulación compara ocho estrategias de planificación diferentes:

- Perfecto. Planificador propuesto en este trabajo y el estimador de los tiempos de procesamiento perfecto (EP).
- Chebyshev. Planificador propuesto en este trabajo y estimador de los tiempos de procesamiento basado en la desigualdad de Chebyshev (EC).
- Media. Planificador propuesto en este trabajo y estimador de los tiempos de procesamiento basado en la media (EM).
- Usuario. Planificador propuesto en este trabajo y estimador de los tiempos de procesamiento que simula a los usuarios (EU).
- Estimador05. Planificador propuesto en este trabajo y estimador de los tiempos de procesamiento basado en el trabajo de Van den Bossche et al [15] y con una precisión de 0,5. En el trabajo de Van den Bossche et al se modela el error en los tiempos de procesamiento de entrada al planificador mediante un factor de exactitud (*acc* del inglés *accuracy*) que toma valores entre 0 y 1. El tiempo de procesamiento con el que se alimenta el planificador se obtiene mediante un generador de números aleatorios basado en una distribución normal cuya media ($\mu$) es el valor exacto del tiempo de procesamiento de la tarea y cuya desviación estándar es $\mu*(1-acc)$. En este trabajo se emplea $acc = 0,5$ por lo que la desviación estándar será $0,5 * \mu$.
- BosschePerfecto. Planificador propuesto por Van den Bossche et al [15] empleando la estrategia descrita como *unfeasible* y una ordenación de la

cola de trabajos mediante EDF. Se emplea el estimador perfecto de los tiempos de procesamiento (EP).

- Bossche05. Planificador propuesto por Van den Bossche et al [15] empleando el estimador propuesto por el mismo autor con $acc = 0,5$ (mismo estimador empleado en Estimador05).
- BosscheChebyshev. Planificador propuesto por Van den Bossche et al [15] empleando el estimador basado en la desigualdad de Chebyshev (EC).

El efecto de producir estimaciones de los tiempos de procesamiento sin contar con muestras de los tiempos de procesamiento de tareas ejecutadas previamente para los estimadores de media y de Chebyshev es eliminado ejecutando previamente simulaciones lo suficientemente largas como para alcanzar el régimen permanente del sistema. La primera de las simulaciones se emplea para obtener un conjunto inicial de tiempos de procesamiento que es posteriormente empleado como entrada en la segunda simulación. De este modo se elimina el régimen transitorio del sistema en el que los estimadores aún no cuentan con suficientes datos. Los resultados presentados se corresponden con los resultados de la segunda simulación, el régimen permanente del sistema.

El proceso de evaluación se puede visualizar gráficamente en la figura 5.1. La evaluación se realiza mediante un simulador de eventos discretos que es alimentado con los diferentes modelos de la carga de trabajo, el modelo de la infraestructura Cloud híbrida a emplear y los parámetros que definen los escenarios de simulación (análisis del impacto de variar el límite temporal de ejecución y análisis de impacto de variar el tiempo entre llegadas de las aplicaciones). El propio simulador durante la ejecución de los escenarios recoge un conjunto de datos que constituyen los resultados de la simulación. Los resultados se presentan de manera gráfica y se realiza un análisis de varianza para conocer si existen diferencias estadísticamente significativas entre los diferentes planificadores.

### 5.1.1. Simulador

Los experimentos de evaluación del algoritmo de planificación se llevaron a cabo mediante un simulador de eventos discretos desarrollado sobre CloudSim [59]. CloudSim es una librería extensible de simulación de eventos discretos desarrollada en Java y que permite modelar y simular entornos de computación en Cloud. El simulador desarrollado se encuentra disponible en un repositorio Git[1] para permitir la repetibilidad de los resultados.

[1] https://gitlab.com/vpelaez-phd/paper-wiley-concurrence-computation.git

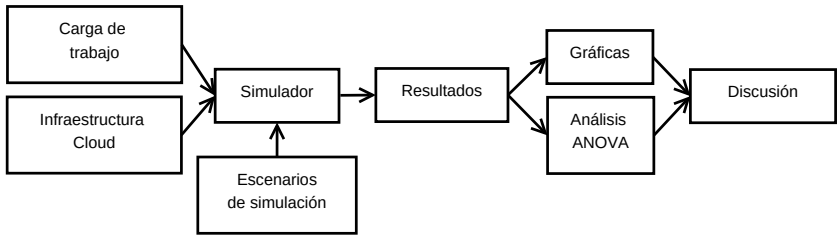



Figura 5.1: Proceso de evaluación, toma de datos y análisis de resultado

El simulador desarrollado esta formado por seis conceptos principales y sus correspondientes entidades: Cloud (entidad *DataCenter*), instancia de máquina virtual (entidad *VM*), aplicación o bolsa de tareas (entidad *Application*), tarea (entidad *Cloudlet*), estimador del tiempo de procesamiento (entidad *RuntimeEstimator*) y planificador (entidad *DataCenterBroker*). Las entidades de tipo *DataCenter* simulan la infraestructura hardware y ejecutan las instancias de máquinas virtuales (entidades *VM*). Las entidades de tipo *VM* simulan la ejecución de las tareas (entidades *Cloudlet*). Las entidades *Cloudlet* representan tareas independientes mientras que la entidad *Application* modela las bolsas de tareas como un conjunto de entidades del tipo *Cloudlet*. Las entidades del tipo *DataCenterBroker* gestionan la planificación de las entidades del tipo *Cloudlet* y el aprovisionamiento de las entidades de tipo *VM* en los diferentes Clouds representados por entidades del tipo *DataCenter*.

Las entidades del tipo *VM*, *Cloudlet* y *DataCenter* extienden las entidades del mismo nombre ya presentes en CloudSim para modificar el comportamiento por defecto y ajustarlo a las necesidades de este trabajo. Por ejemplo, las entidades de tipo *VM* desarrolladas añaden la posibilidad de simular el retardo en el aprovisionamiento de las instancias de máquinas virtuales. Las entidades de tipo *Cloudlet* y *Application* añaden el comportamiento necesario para modelar el límite temporal de ejecución y la entidad de tipo *DataCenter* incluye la funcionalidad necesaria para aprovisionar instancias de máquinas virtuales de manera dinámica. El Cloud privado y cada uno de los Clouds públicos son modelados mediante entidades del tipo *DataCenter*.

Las entidades de tipo *DataCenterBroker* son la parte central del simulador y son las encargadas de simular la llegada de aplicaciones de acuerdo a la especificación de la carga de trabajo, la implementación de los algoritmos de planificación y la gestión de las instancias de máquinas virtuales. Los dos sub-planificadores que componen el algoritmo de planificación propuesto en es-

te trabajo (el planificador del Cloud privado y el planificador del Cloud público) se implementan empleando este tipo de entidad. La cola de aplicaciones, la planificación actual y el estado de las instancias de máquinas virtuales en ejecución se mantienen dentro de esta entidad. La entidad *DataCenterBroker* representa el software que debería ser incluido en un sistema real para implementar el planificador propuesto.

Aunque CloudSim permite simular por completo todos los aspectos de una infraestructura Cloud, en este trabajo no se emplean todas sus funcionalidades dado que este está centrado en la ejecución de tareas en instancias de máquinas virtuales y la planificación y aprovisionamiento a alto nivel. Por ello se emplean las políticas simples ya incluidas en CloudSim para simular la infraestructura hardware:

- Se emplea la política *VmAllocationPolicySimple* para seleccionar el servidor físico que alberga cada instancia de máquina virtual. Esta política asigna la instancia al servidor con menos procesadores en uso.
- Se emplea la política *VmSchedulerTimeShared* para repartir las CPUs de un servidor entre las instancias de máquinas virtuales asignadas a dicho servidor. Esta política permite a las instancias de máquinas virtuales compartir el mismo procesador en un servidor.
- Se emplea la política *CloudletSchedulerTimeShared* para distribuir la ejecución de las tareas dentro de una instancia de máquina virtual. Esta política permite a las tareas ejecutarse compartiendo el tiempo de CPU.

La especificación de la carga de trabajo, los Clouds disponibles, los tipos de máquinas virtuales y los tiempos de aprovisionamiento de las instancias son leídas por el simulador de ficheros CSV (fichero de valores separados por comas) previamente generados. Estos ficheros con los datos de los escenarios de simulación han sido creados de acuerdo a las especificaciones presentadas en los siguientes apartados.

### 5.1.2. Entorno Cloud

Con el objetivo de llevar a cabo un análisis lo más general posible, la infraestructura de Cloud público empleada en la evaluación del planificador se modela empleando dos proveedores reales: Amazon EC2 y Google Compute Engine.

Uno de los principales problemas cuando se modelan entornos de Cloud empleando CloudSim es la definición de la velocidad de procesamiento de las CPUs

en las instancias de máquinas virtuales. CloudSim necesita que las velocidades de las CPUs se especifiquen en millones de instrucciones por segundo (del inglés *Millions of Instructions per Second - MIPS*) y que el tamaño de las tareas se exprese en millones de instrucciones. Sin embargo, los proveedores de Cloud públicos no miden la velocidad en MIPS o en otras métricas estándar como FLOPS (*Floating point operations per Second*). Tanto Amazon como Google proporcionan sus propias unidades de medida de la velocidad de CPU: *EC2 Compute Unit* (*ECU*) en el caso de Amazon y *Google Compute Engine Units* (*GCEU*) en el caso de Google. Esta divergencia a la hora de especificar el rendimiento de los recursos computacionales de un proveedor a otro hace muy difícil la construcción de modelos precisos para simular entornos reales. Por esta razón existen muchos trabajos y estudios como los de Sarda et al[60] o cmips.net[2] que pretenden comparar el rendimiento de las máquinas virtuales de diferentes proveedores.

En otros estudios en el estado del arte se emplean los ECU y el número de CPUs virtuales (vCPUS) como base para asignar los MIPS de cada procesador. En el trabajo de Calheiros and Buyya [11] la velocidad de cada núcleo de ejecución se establece dividiendo el número de ECUs entre el número de CPUs virtuales indicado por el proveedor de Cloud. Aunque este sistema puede no ser un reflejo exacto de la realidad, sirve para establecer una base común para comparar las velocidades de las CPUs. Desde el punto de vista de la simulación este enfoque es válido aunque no es preciso si se tienen en cuenta los resultados de *benchmarks* reales[60]. En el trabajo de Van den Bossche et al[15], todas las máquinas virtuales se consideran iguales en términos de la velocidad de CPU y por tanto el problema de asignar diferentes velocidades en función del tipo de máquina virtual o proveedor no es analizado. En este estudio se asume que cada tipo de máquina virtual puede tener una velocidad de CPU diferente que es desconocida desde el punto de vista del planificador. Para asignar los MIPS a cada tipo de máquina virtual se considera la información proporcionada por el proveedor de Cloud (el número de vCPUS) y los benchmarks de cmips.net.

La tabla 5.1 muestra los tipos de máquinas virtuales de Amazon EC2 considerados en este trabajo y que se corresponden con la categoría de "Instancias bajo demanda - Linux - Propósito general". Una unidad de computación EC2 equivale a la capacidad de una CPU Opteron del año 2007 o un procesador Xeon del año 2007 con una frecuencia de reloj de 1.0 a 1.2 GHz. De acuerdo a la información proporcionada por Amazon y los *benchmarks* de cmips.net se asigna un valor constante de 160 MIPS por cada ECU.

[2] http://www.cmips.net

| Nombre | vCPU | Velocidad CPU (ECU) | RAM (GB) | Precio por hora | MIPS |
|---|---|---|---|---|---|
| m4.large | 2 | 3.25 | 8 | $0.126 | 520 |
| m4.xlarge | 4 | 3.25 | 16 | $0.252 | 520 |
| m4.2xlarge | 8 | 3.25 | 32 | $0.504 | 520 |
| m4.4xlarge | 16 | 3.43 | 64 | $1.008 | 548 |
| m4.10xlarge | 40 | 3.11 | 160 | $2.52 | 497 |
| m3.medium | 1 | 3 | 3.75 | $0.067 | 480 |
| m3.large | 2 | 3.25 | 7.5 | $0.133 | 520 |
| m3.xlarge | 4 | 3.25 | 15 | $0.266 | 520 |
| m3.2xlarge | 8 | 3.25 | 30 | $0.532 | 520 |

Tabla 5.1: Instancias de Amazon EC2

| Nombre | vCPU | Velocidad CPU (GCEU) | RAM (GB) | Precio por hora | MIPS |
|---|---|---|---|---|---|
| g1-small | 1 | 1.38 | 1.70 | $0.027 | 263 |
| n1-standard-1 | 1 | 2.75 | 3.75 | $0.050 | 580 |
| n1-standard-2 | 2 | 2.75 | 7.50 | $0.100 | 580 |
| n1-standard-4 | 4 | 2.75 | 15 | $0.200 | 580 |
| n1-standard-8 | 8 | 2.75 | 30 | $0.400 | 580 |
| n1-standard-16 | 16 | 2.75 | 60 | $0.800 | 580 |
| n1-standard-32 | 32 | 2.75 | 120 | $1.600 | 580 |

Tabla 5.2: Instancias de Google Compute Engine

La tabla 5.2 muestra los tipos de máquinas virtuales de Google Compute Engine empleados en este trabajo y que se corresponden con los tipos de la categoría "Tipos de máquinas estándar". Google emplea 2,75 GCEUs para representar la mínima capacidad de computación de una CPU virtual y que según su especificación se corresponde con un hyper-thread hardware en procesadores Intel Sandry Bridge, Ivy Bridge o Haswell. Un GCEU es al menos tan potente como la capacidad de computación de una CPU Opteron de 2007 con una frecuencia de 1.0 a 1.2 GHz. Teniendo en cuenta esta información y los *benchmarks* de cmips.net se asigna un valor constante de 210 MIPS por GCEU para las instancias estándar y de 263 MIPS para la instancia G1-small.

El modelo de simulación asume que el Cloud privado reside en la infraestructura hardware del proveedor de servicio y está compuesto por 50 máquinas

virtuales de 4 CPUs con 550 MIPS y un memoria RAM de 8GB. El valor de MIPS para las CPUs de las máquinas virtuales del Cloud privado se ha escogido de tal manera que sea mayor que el de las máquinas virtuales de Amazon EC2 pero menor que el de las máquinas de Google Compute Engine. De esta forma, en el Cloud público se dispone de máquinas tanto más rápidas como más lentas que las del Cloud privado, lo que permite generar escenarios de evaluación más generales.

### 5.1.3. Cargas de trabajo

En este trabajo se emplean tres cargas de trabajo diferentes para poder validar el planificador en diferentes condiciones. La primera carga de trabajo ha sido generada sintéticamente usando un modelo derivado de una carga real, la segunda carga ha sido generada de manera sintética siguiendo la metodología de otros trabajos y la última carga esta basada en la traza *Google Cluster Data* y denominada clusterdata-2011-2[61].

Las dos primeras cargas sintéticas han sido generadas empleando un modelo con siete parámetros de manera similar a como se hace en [15]. El tamaño de las cargas ronda las 19000 tareas y abarca un periodo de simulación de aproximadamente 4 días con la tasa de llegada de aplicaciones más baja. Estas cifras se han seleccionado para llevar el Cloud privado a su punto de saturación y justificar el uso de un Cloud híbrido. Los siete parámetros del modelo son:

1. Número de usuarios en la carga de trabajo. Cada carga de trabajo contiene aplicaciones de 10 usuarios diferentes.

2. Número de aplicaciones enviadas para ejecución por cada usuario. Cada usuario envía 75 aplicaciones.

3. Tasa de llegada de las aplicaciones. Las aplicaciones llegan al sistema cada segundo siguiendo un distribución de Poisson. Con el objetivo de evaluar el planificador bajo diferentes condiciones la tasa de llegada de aplicaciones se varía desde una tasa que sigue una Poisson con $\lambda = 0,002$ hasta una tasa de llegada que sigue una Poisson con $\lambda = 8$. Esto equivale a unas tasas de llegada que varían desde 0.002 aplicaciones por segundo hasta 8 aplicaciones por segundo.

4. Trabajos (tareas) por cada aplicación. El número de trabajos dentro de una aplicación es aleatorio siguiendo una función de distribución uniforme entre 1 y 50.

5. Tiempo de procesamiento base por usuario. Como se vio en la sección 4.3, los tiempo de procesamiento de los trabajos de una aplicación del mismo usuario están típicamente relacionados y suelen aproximarse a un tiempo de procesamiento común que aquí denominaremos el tiempo de procesamiento base del usuario. Dado que la manera de modelar los tiempos de procesamiento puede afectar a las estimaciones de dichos tiempos, en este trabajo se modelan los tiempos de procesamiento de las tareas que llegan al sistema mediante funciones de distribución de probabilidad sobre la variable aleatoria y continua que representaría el tiempo de procesamiento. El tiempo de procesamiento base de cada usuario se obtiene mediante un generador de números aleatorios que emplea la función de distribución elegida para la generación. Estas distribuciones controlan por tanto los tiempos de ejecución en general de todos los trabajos que llegan al sistema.

6. Tiempo de procesamiento de cada trabajo (tarea). El tiempo de procesamiento de cada trabajo individual se obtiene mediante un generador de números aleatorios basado en una distribución Normal que emplea como media el tiempo de procesamiento base del usuario y como desviación estándar el 50 % de la media. Se emplea una distribución normal de manera similar a como hace Van den Bossche et al [15] y en base a los resultados de Iosup et al [62] que sugieren que el tiempo de procesamiento de las tareas dentro de una bolsa de tareas sigue una distribución normal.

7. Factor de límite temporal de ejecución de las aplicaciones. Dado que el límite temporal de ejecución condiciona el coste total de la infraestructura pública empleada y el comportamiento del algoritmo, el límite temporal de ejecución de las aplicaciones se modela como un factor del tiempo de procesamiento base del usuario en el tipo de máquina virtual más rápida. Para poder cubrir diferentes escenarios el factor se varia entre 2 y 9 para cada usuario. Es decir, un factor de 2 fija el límite temporal de ejecución de una aplicación en dos veces el tiempo base de procesamiento del usuario al que pertenezca la aplicación.

Las cargas sintéticas se diferencian tan solo en la función de distribución de los tiempos de procesamiento empleadas, siendo una Weibull y una Normal:

- El modelo basado en una distribución Weibull se basa en la carga de trabajo sintética presentada por Van den Bossche et al[15], donde los parámetros

| Parámetro | Modelo Weibull | Modelo Normal |
|---|---|---|
| Número de usuarios | 10 | 10 |
| Aplicaciones por usuario | 75 | 75 |
| Tasa de llegada de aplicaciones | Poisson(0.002)<br>Poisson(8) | Poisson(0.002)<br>Poisson(8) |
| Tareas por aplicación | Uniform(50) | Uniform(50) |
| Tiempo de procesamiento base de los usuario ($b$) | Weibull<br>$\lambda = 1879(scale)$<br>$\beta = 0{,}426(shape)$ | Normal<br>$\mu = 1879$<br>$\sigma = 939$ |
| Tiempo de procesamiento de las tareas de un usuario con tiempo base $b$ | Normal($b$, $0,5 \cdot b$) | Normal($b$, $0,5 \cdot b$) |
| Factor de límite temporal de ejecución | 2 a 9 | 2 a 9 |

Tabla 5.3: Parámetros de las cargas de trabajo sintéticas

de la distribución se derivan de la traza de la carga de trabajo del *Parallel Workloads Archive's SDSC IBM SP2*[3].

- Los parámetros del modelo basado en una distribución Normal toman valores escogidos arbitrariamente. El uso de la distribución normal para modelar el tiempo de procesamiento, y en general cualquier otro parámetro de una carga, es una práctica habitual en el estado del arte. Wu et al [31] modela los tiempos de procesamiento de un usuario como una normal que toma como media un tiempo de procesamiento que varía en función de cinco categorías y una desviación estándar de 0,5 veces la media. Calheiros et al [59] también emplean una distribución normal para generar tiempos de procesamiento aleatorios aunque sin especificar sus parámetros. Oprescu et al [63] emplea una estrategia similar generando los tiempos de procesamiento de las tareas mediante una distribución normal $N(15, \sigma), \sigma = \sqrt{5}$.

La tabla 5.3 resume el valor de los diferentes parámetros de las cargas de trabajo sintéticas.

La traza clusterdata-2011-2 proporcionada por Google contiene información de 29 días del mes de mayo de 2011 en un cluster real de Google con aproximadamente 12500 máquinas. El principal problema para emplear esta traza es

[3] http://www.cs.huji.ac.il/labs/parallel/workload/

que ha sido publicada con datos ofuscados para salvaguardar información confidencial del negocio de Google [64]. La traza contiene información acerca de las máquinas del cluster y de la carga de trabajo atendida por dichas máquinas. La carga de trabajo presente en la traza se compone de 37516956 tareas agrupadas en 672074 trabajos que se corresponden con 39730 aplicaciones enviadas a ejecución por 933 usuarios. Las aplicaciones se pueden clasificar en cuatro categorías diferentes: aplicaciones de una sola tarea, aplicaciones con tareas secuenciales, aplicaciones en modo *batch* y aplicaciones mixtas [65]. Las características detalladas de la carga de trabajo han sido previamente estudiadas por otros autores [65] [66].

Desde el punto de vista de este trabajo un usuario en la traza de Google es la combinación de una aplicación y un usuario dado que cada aplicación de un usuario debería tener un comportamiento similar. El siguiente procedimiento ha sido empleado para generar una carga de entrada válida como entrada al simulador de este trabajo a partir de la traza de Google:

1. La información relativa a la máquinas del cluster se ha empleado para agrupar las máquinas en diferentes grupos. La traza contiene para cada máquina un conjunto de atributos y un conjunto de eventos relacionados con sus capacidades. Para realizar la agrupación de máquinas en categorías se han empleado solo aquellos atributos con más de un valor diferente y con menos de 100 valores diferentes (algunos atributos toman un valor diferente por cada máquina, por ejemplo la MAC, y otros atributos toman el mismo valor para todas las máquinas). Mediante este procedimiento las 12583 máquinas presentes en la traza se agruparon en 634 grupos diferentes en base a 24 de los 67 atributos. Estos 634 grupos constituyen tipos de máquinas desde el punto de vista de la ejecución de los trabajos.

2. Las aplicaciones y los trabajos son filtrados para considerar solo las aplicaciones que tienen una estructura de tipo *batch*, siendo este tipo el que se corresponde con una carga de tipo de bolsas de tareas, y siempre compuestas por más de un trabajo. Este filtrado redujo la carga de trabajo en la traza a 1472 aplicaciones compuestas por 8266 trabajos.

3. Para poder obtener tiempos de procesamiento de los trabajos comparables entre si se filtraron los trabajos para dejar solo los trabajos que se ejecutasen sobre el tipo de máquina más común de los trabajos resultantes del filtrado en la etapa anterior. Esto redujo la traza a 291 aplicaciones de 56 usuarios diferentes y con un total de 1634 trabajos. Al realizar la conversión entre aplicaciones y usuarios de Google a las aplicaciones y usuarios

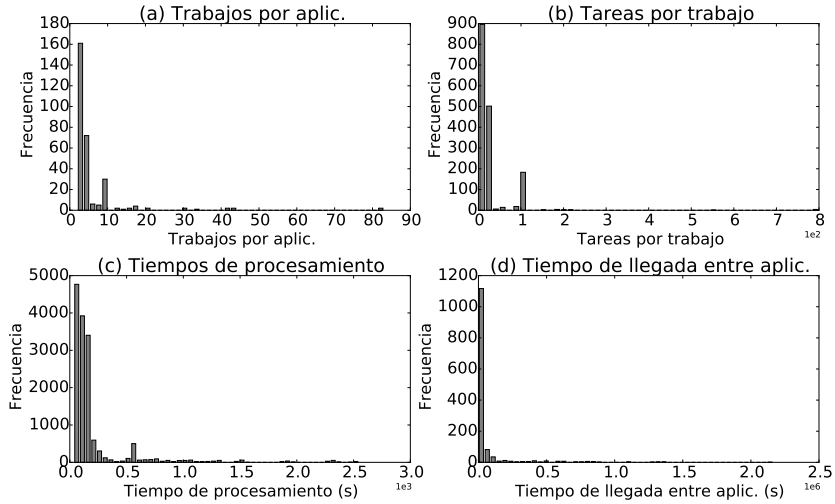



Figura 5.2: Caracterización de la carga de trabajo basada en la traza de Google: A. Trabajos por aplicación, B. Tareas por trabajo, C. Tiempos de procesamiento, D. Tiempo entre llegadas.

considerados en este trabajo la carga final se compone de 1634 trabajos pertenecientes a 294 usuarios (294 parejas usuario-aplicación en la carga de Google).

La figura 5.2 muestra las características de la carga de trabajo resultante del proceso de filtrado realizado sobre la traza original de Google. Aunque las distribuciones de frecuencia muestran formas de distribuciones bien definidas, se debe tener en cuenta que cada usuario tiene su propia distribución de los tiempos de procesamiento de los trabajos y una tasa de llegada de aplicaciones también diferente. Los percentiles 25, 50 y 75 de los tiempos de procesamiento de los trabajos son 74, 127 y 161 segundos respectivamente. Los percentiles 25, 50 y 75 de los tiempos entre llegadas de las aplicaciones son 405, 892 y 3786 segundos respectivamente.

### 5.1.4. Escenarios de simulación

Los escenarios de simulación se obtienen de combinar los siguientes elementos:

- Objetivo del escenario: analizar el impacto de variar el límite temporal o analizar el impacto de variar el ratio de llegada de aplicaciones al sistema.
- Tipo de carga: carga sintética basada en la distribución Weibull, carga sintética basada en la distribución Normal o carga de Google.
- Tiempo de aprovisionamiento de instancias de máquinas virtuales: ideal (aprovisionamiento inmediato, espera de 0 segundos) o aprovisionamiento con retardo de 100 segundos.

Mediante la combinación de valores de los tres elementos anteriores se obtienen los ocho escenarios de simulación propuestos en este trabajo:

- Análisis por límite temporal, carga Weibull, tiempo de aprovisionamiento de 0 segundos.
- Análisis por límite temporal, carga Weibull, tiempo de aprovisionamiento de 100 segundos.
- Análisis por límite temporal, carga Normal, tiempo de aprovisionamiento de 0 segundos.
- Análisis por límite temporal, carga Normal, tiempo de aprovisionamiento de 100 segundos.
- Análisis por límite temporal, carga de Google, tiempo de aprovisionamiento de 0 segundos.
- Análisis por ratio de llegada de aplicaciones, carga Weibull, tiempo de aprovisionamiento de 0 segundos.
- Análisis por ratio de llegada de aplicaciones, carga Weibull, tiempo de aprovisionamiento de 100 segundos.
- Análisis por ratio de llegada de aplicaciones, carga Normal, tiempo de aprovisionamiento de 100 segundos.

### 5.1.5. Datos resultantes de cada simulación

Para cada simulación los resultados se presentan mediante cinco figuras en las que existe una gráfica por cada combinación de planificador y estimador (estrategia de planificación). Las figuras hacen referencia a cinco indicadores diferentes de la calidad de la planificación producida:

1. Aplicaciones en plazo: porcentaje de aplicaciones (bolsas de tareas) cuya ejecución ha finalizado antes de su límite temporal de ejecución.

2. Aplicaciones fuera de plazo: porcentaje de aplicaciones cuya ejecución ha finalizado después de su límite temporal de ejecución. El porcentaje de las aplicaciones plazo más el porcentaje de aplicaciones fuera de plazo más el porcentaje de aplicaciones inviables es 100. Las aplicaciones inviables son aquellas que no han sido ejecutadas porque el planificador ha considerado que no era posible su ejecución en el plazo establecido por el límite temporal. El porcentaje de aplicaciones inviables se obtiene de restar a 100 el porcentaje de aplicaciones en plazo más el porcentaje de aplicaciones fuera de plazo.

3. Tiempo de espera: media del tiempo que un trabajo espera en las colas del planificador desde su llegada al sistema y hasta que comienza su ejecución.

4. Coste por aplicación en plazo: media del coste por cada aplicación ejecutada en plazo. El coste total depende del número de instancias de máquinas virtuales aprovisionadas y del tiempo que dichas instancias se encuentran en ejecución.

5. VM iniciadas: número de instancias de máquinas virtuales que se han iniciado para poder ejecutar la carga de trabajo.

### 5.1.6. Análisis de varianza (ANOVA)

Los datos resultantes de ejecutar la simulación para cada uno los ocho escenarios de simulación propuestos pueden ser caracterizados mediante tuplas formadas por las siguientes variables:

1. Tipo de análisis: por límite temporal o por ratio de llegada de aplicaciones

2. Carga de trabajo: Weibull, Normal o Google

3. Tiempo de aprovisionamiento: 0 ó 100 segundos.

4. Factor: valor del límite temporal o del ratio de llegada de aplicaciones

5. Algoritmo: propuesto en este trabajo o propuesto por Van den Bossche

6. Estimador: Perfecto, Chebyshev, Media, Usuario, Estimador05

7. Porcentaje de aplicaciones ejecutadas en plazo

8. Porcentaje de aplicaciones ejecutadas fuera de plazo
9. Tiempo medio de espera
10. Coste por aplicación en plazo
11. Número de instancias de máquinas virtuales iniciadas.

Las variables 1 a 6 son las variables independientes del experimento desde el punto de vista del análisis estadístico, mientras que las variables 7 a 11 son las variables dependientes. Además, las variables independientes son todas de tipo categórico, dado que solo pueden tomar un conjunto de valores finitos.

El objetivo del análisis de varianza que se lleva a cabo en este trabajo es conocer qué variables independientes tienen un efecto estadísticamente significativo sobre los resultados (variables dependientes) y si las diferencias entre unos y otros escenarios son también estadísticamente significativas [67]. El objetivo principal es determinar si el algoritmo o los estimadores de los tiempos de procesamiento son estadísticamente determinantes en las diferencias que se obtengan en los diferentes escenarios de simulación.

Dado que se tienen 6 variables independientes o explicativas de los resultados de cada experimento (simulación), estaríamos ante un caso de ANOVA con 6 factores (*six-way ANOVA*). Además, aunque existen cuatro variables dependientes, el análisis se realiza por cada variable independiente de manera individual, dado que si no la complejidad del modelo sería excesiva. Dado que cada combinación de niveles (valores) de las variables independientes no tiene replicación en las simulaciones, no es posible estudiar la interacción entre los factores (variables independientes), es decir, el diseño del experimento hace que no sea posible un ANOVA factorial. Además el alto número de variables independientes también haría muy compleja la interpretación de las interacciones.

El modelo general ANOVA empleado en este trabajo se podría expresar empleando la sintaxis del módulo Formulae de R [68] (a su vez basado en el lenguaje de fórmulas de S [69]) según la ecuación 5.1

$$\begin{aligned} enplazo \sim\ & C(analisis) + C(carga) + C(aprovisionamiento) \\ & + C(factor) + C(algoritmo) + C(estimador) \end{aligned} \quad (5.1)$$

La variable *enplazo* representa el porcentaje de aplicaciones ejecutadas en plazo y debe tenerse en cuenta que el modelo general ANOVA presentado en la ecuación se repite por cada una de las variables independientes del experimento.

Cabe notar que aunque se podría aplicar un análisis de varianza multivariado que analizase todas las variables dependientes en conjunto (MANOVA), la forma habitual de proceder es la realización de análisis independientes para facilitar su interpretación. Este es el enfoque empleado en este trabajo.

Sin embargo, dado que los dos tipos de análisis se consideran independientes al responder a cuestiones de investigación independientes, el conjunto de simulaciones se divide en dos y nunca se comparan o analizan los resultados de los dos tipos de análisis en conjunto. Esto hace que la variable tipo de análisis (análisis) pueda ser eliminada como variable independiente y considerar cada tipo de esta variable como un experimento diferente. De este modo el modelo ANOVA empleado se presenta en la ecuación 5.2.

$$enplazo \sim\ C(carga) + C(aprovisionamiento) + C(factor) + C(algoritmo) + C(estimador) \quad (5.2)$$

Por la manera en la que se organizan las simulaciones en escenarios, cada escenario fija el tipo de carga de trabajo y el tiempo de aprovisionamiento de nuevas instancias de máquinas virtuales. Esto hace que por cada simulación el modelo ANOVA pueda reducirse aún más, eliminado estas dos variables independientes y obteniendo el modelo de la ecuación 5.3. En cada simulación se obtienen resultados que varían en cuanto al factor (límite temporal o tasa de llegada de aplicaciones), el algoritmo de planificación empleado y el estimador de los tiempos de procesamiento. El resto de variables independientes se encuentran fijadas por la propia definición del escenario que se esta simulando.

$$enplazo \sim\ C(factor) + C(algoritmo) + C(estimador) \quad (5.3)$$

Aunque cada simulación independiente es analizada según el modelo de la ecuación 5.3, los resultados de varias simulaciones para el mismo tipo de análisis se agrupan para llevar a cabo un análisis que permitan conocer el impacto del tipo de carga de trabajo y del tiempo de aprovisionamiento. En estos análisis mas generales se emplea el modelo ANOVA de la ecuación 5.2.

Los supuestos o precondiciones para un análisis de varianza que se comprueban en este trabajo y el test empleado para su comprobación se detallan a continuación:

1. Normalidad: La variable dependiente se aproxima a una distribución normal. En nuestro análisis se verifica mediante el test de Saphiro.

2. Homocedasticidad: Los residuos siguen una distribución normal, también denominado homogeneidad de varianza. Se verifica mediante el test Omnibus.

3. No multicolinealidad: Se verifica mediante el número de condicionalidad.

Cabe destacar que en todos los casos la normalidad no se cumple, del mismo modo que la homocedasticidad en la gran mayoría de los mismos. Sin embargo estas dos condiciones pueden ser obviadas cuando el número de niveles de los factores se encuentra equilibrado, como en el caso de este trabajo. En este caso el estadístico F, inherente al análisis ANOVA, es resistente a violaciones de la normalidad y la homocedasticidad [70]. En nuestro caso todos los niveles de los grupos están equilibrados ya que se realiza el mismo número de simulaciones por cada nivel de un factor (mismo número de simulaciones por cada algoritmo y estimador). La condición de no multicolinealidad se verifica en todos los análisis realizados.

Para que las diferencias sean consideradas estadísticamente significativas el valor $p$ debe ser inferior a 0,05, rechazando en ese caso la hipótesis nula y pudiendo afirmar que las diferencias entre las medias de los diferentes grupos son estadísticamente significativas. Aunque un factor sea estadísticamente significativo con respecto a los resultados, puede que su efecto sea muy pequeño o limitado, para ello se presenta además el tamaño del efecto mediante $\omega^2$ (omega al cuadrado).

Cuando se encuentran diferencias estadísticamente significativas para una variable independiente se realiza un análisis posterior (*post-hoc analysis*) con el test Tukey's HSD (del inglés *honestly significant difference*) [71] para determinar que valores (niveles) de la variable independiente generan las diferencias estadísticamente significativas en la media de la variable dependiente. Esto permite asegurar que los resultados del test ANOVA inicial son correctos y determinar los valores que tienen influencia en las diferencias.

## 5.2. Análisis por límite temporal

En estas simulaciones la tasa de llegada de aplicaciones está fija siguiendo una distribución Poisson(0,002) mientras que el factor del límite temporal varía de 2 a 9. En el caso de la carga de trabajo basada en los datos del cluster de Google la tasa de llegada de las aplicaciones no se ha modificado y es la que estaba presente en la traza, mientras que el factor del límite temporal de ejecución se varía de 2 a 6.

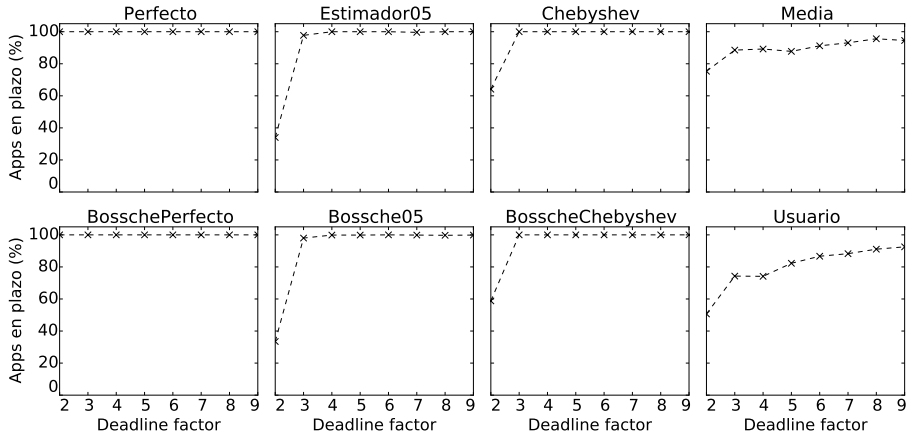



Figura 5.3: Carga Weibull, análisis límite temporal, tiempo de aprovisionamiento 0 segundos, porcentaje de aplicaciones ejecutadas en plazo

### 5.2.1. Carga Weibull con tiempo de aprovisionamiento ideal

Las figuras 5.3, 5.4, 5.5, 5.6 y 5.7 muestran los resultados de la simulación con la carga de trabajo modelada mediante una distribución Weibull y con tiempos de aprovisionamiento de las instancias de máquinas virtuales ideales ($avt_{cti} - rqt_{cti} = 0$). La tabla B.1 del anexo B contiene los resultados numéricos.

Como se puede apreciar en la figura 5.3 tanto el planificador propuesto en este trabajo como en el trabajo de Van den Bossche obtienen los mejores resultados en términos de trabajos ejecutados en plazo cuando se emplea el estimador perfecto. Ambos planificadores obtienen un 100 % de aplicaciones ejecutadas en plazo en un sistema con capacidad ilimitada (número ilimitado de instancias de máquinas virtuales) e instancias de máquinas virtuales que son aprovisionadas instantáneamente.

Cuando los planificadores emplean el estimador basado en la desigualdad de Chebyshev los resultados obtenidos son muy similares a los del estimador perfecto para factores de límite temporal mayores de 2. Sin embargo con un factor de límite temporal igual a 2 las estimaciones producidas por el estimador de Chebyshev son demasiado pesimistas y el número de aplicaciones ejecutadas en plazo es bajo. Estas estimaciones pesimistas son mayores que el tiempo disponible para la ejecución de las aplicaciones antes de su límite temporal de ejecución en la instancia más rápida de máquina virtual. En este escenario, el

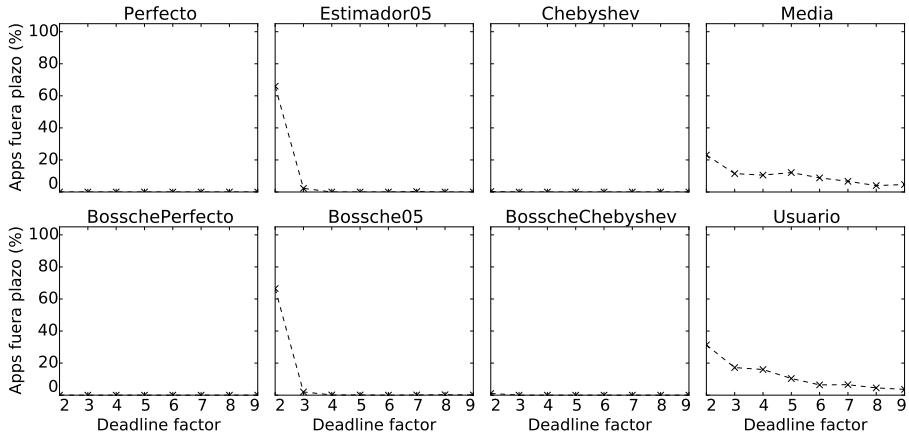



Figura 5.4: Carga Weibull, análisis límite temporal, tiempo de aprovisionamiento 0 segundos, porcentaje de aplicaciones ejecutadas fuera de plazo

planificador marca esas aplicaciones como inviables. El estimador basado en la desigualdad de Chebyshev produce estimaciones pesimistas en el sentido de que sobreestiman los tiempos de procesamiento, lo que hace que muy pocas aplicaciones que se envían a ejecución no cumplan el límite temporal, al contrario de lo que ocurre con otros estimadores como se muestra en la figura 5.4.

El análisis ANOVA para las aplicaciones ejecutadas en plazo indica que el factor de límite temporal (el tamaño del efecto es del 48 % - $\omega^2 = 0{,}484$) y el estimador del tiempo de procesamiento (el tamaño del efecto es del 12 % - $\omega^2 = 0{,}125$) tienen una influencia estadísticamente significativa en las diferencias de los resultados. El análisis post-hoc indica que existen diferencias estadísticamente significativas entre el valor de límite temporal 2 y el resto de valores y entre el estimador Perfecto y el estimador de Usuario.

El análisis ANOVA para las aplicaciones ejecutadas fuera de plazo indica que el factor de límite temporal (el tamaño del efecto es del 25 % - $\omega^2 = 0{,}250$) y el estimador del tiempo de procesamiento (el tamaño del efecto es del 10 % - $\omega^2 = 0{,}103$) tienen una influencia estadísticamente significativa en las diferencias de los resultados. El análisis post-hoc indica que existen diferencias estadísticamente significativas entre el valor de límite temporal 2 y el resto de valores y no se encuentran diferencias entre los estimadores.

Respecto al tiempo medio de espera de los trabajos en el sistema, la figura 5.5

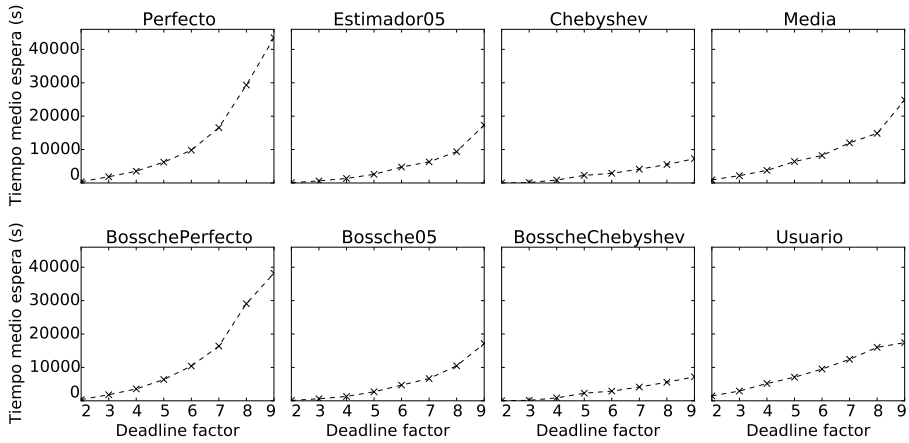



Figura 5.5: Carga Weibull, análisis límite temporal, tiempo de aprovisionamiento 0 segundos, tiempo medio de espera por aplicación

muestra que el tiempo de espera se incrementa a medida que el factor del límite temporal se incrementa. Cuando se emplea el estimador perfecto se obtienen los mayores tiempos de espera debido a que el algoritmo de planificación conoce sin error la holgura disponible para ejecutar cada aplicación.

El análisis ANOVA para el tiempo medio de espera indica que el factor de límite temporal (el tamaño del efecto es del 55 % - $\omega^2 = 0{,}552$) y el estimador del tiempo de procesamiento (el tamaño del efecto es del 17 % - $\omega^2 = 0{,}178$) tienen una influencia estadísticamente significativa en las diferencias de los resultados. El análisis post-hoc indica que existen diferencias estadísticamente significativas entre los valores de límite temporal 8 y 9 y el resto de valores y entre los estimadores Perfecto y de Chebyshev. Este último resultado esta acorde a lo mostrado en la figura 5.5, donde el estimador de Chebyshev tiene tiempos de espera menores que el resto de estimadores y en especial menores que los obtenidos por el estimador Perfecto. Esto se debe a la sobreestimación del estimador de Chebyshev, que hace que las tareas pasen menos tiempo en la cola de aplicaciones.

Como se puede observar en la figura 5.6 el coste por aplicación ejecutada en plazo de las planificaciones generadas es muy similar en todos los casos, obteniendo el estimador perfecto los costes más bajos. El análisis ANOVA para el coste por aplicación indica que el factor de límite temporal (el tamaño del

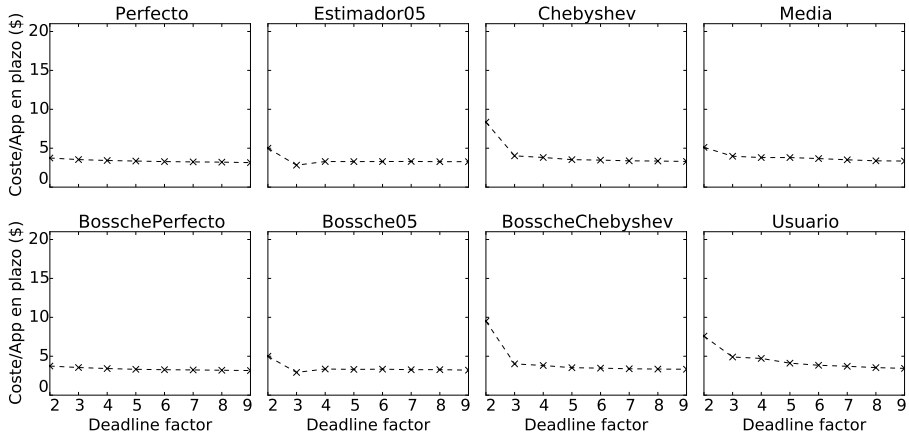



Figura 5.6: Carga Weibull, análisis límite temporal, tiempo de aprovisionamiento 0 segundos, coste por aplicación ejecutada en plazo

efecto es del 49 % - $\omega^2 = 0{,}493$) y el estimador del tiempo de procesamiento (el tamaño del efecto es del 10 % - $\omega^2 = 0{,}178$) tienen una influencia estadísticamente significativa en las diferencias de los resultados. El análisis post-hoc indica que existen diferencias estadísticamente significativas entre el valor de límite temporal 2 y el resto de valores, pero no encuentra diferencias entre los estimadores.

Cabe destacar, como se muestra en la figura 5.7, que cuando se emplea el estimador de Chebyshev se inicia el mayor número de instancias de máquinas virtuales, aunque como se vio en la figura 5.6 los costes por aplicación ejecutada en plazo son similares en todos los casos. El coste no está solo relacionado con el número de instancias de máquinas virtuales iniciadas, sino con el tiempo que dichas instancias están operativas. El análisis ANOVA para el número de instancias de máquinas virtuales indica que el factor de límite temporal (el tamaño del efecto es del 35 % - $\omega^2 = 0{,}351$) y el estimador del tiempo de procesamiento (el tamaño del efecto es del 12 % - $\omega^2 = 0{,}122$) tienen una influencia estadísticamente significativa en las diferencias de los resultados. El análisis post-hoc indica que existen diferencias estadísticamente significativas entre el valor de límite temporal 2 y el resto de valores, y entre el estimador de Chebyshev y el Estimador05.

En general el planificador propuesto en este trabajo obtiene unos resultados

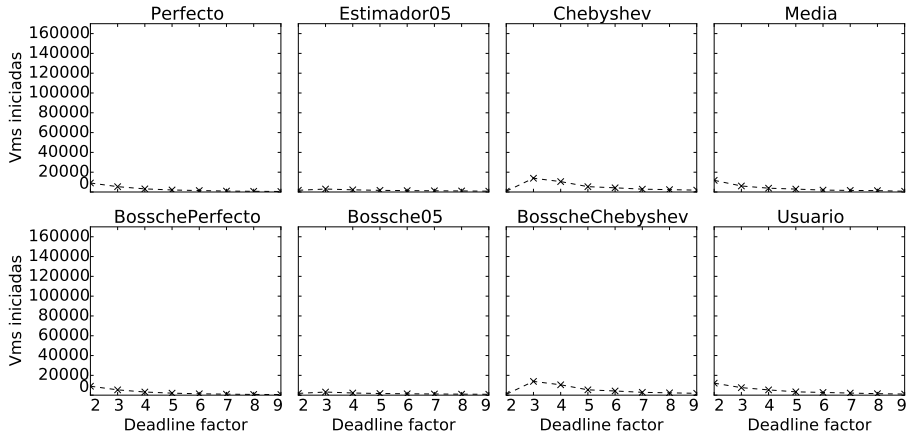



Figura 5.7: Carga Weibull, análisis límite temporal, tiempo de aprovisionamiento 0 segundos, instancias de máquinas virtuales iniciadas

similares a los del planificador propuesto por Van den Bossche cuando se emplea el mismo estimador de los tiempos de procesamiento. Cuando el factor de límite temporal es de 2, nuestro planificador obtiene resultados ligeramente mejores en términos de aplicaciones ejecutadas en plazo y de coste. En los análisis ANOVA realizados para todas las variables independientes no se han podido confirmar diferencias estadísticamente significativas entre ambos algoritmos, confirmando que los resultados son similares desde el punto de vista de los datos. El aspecto que más influye en los resultados es el factor de límite temporal, independientemente del algoritmo de planificación o del estimador empleado.

### 5.2.2. Carga Weibull con tiempo de aprovisionamiento de 100 segundos

Las figuras 5.8, 5.9, 5.10, 5.11 y 5.12 muestran los resultados de la simulación con la carga de trabajo modelada mediante una distribución Weibull y con tiempos de aprovisionamiento de las instancias de máquinas virtuales constantes de 100 segundos ($avt_{cti} - rqt_{cti} = 100$). La tabla B.2 del anexo B contiene los resultados numéricos.

Como se muestra en la figura 5.8 los resultados en términos de aplicaciones ejecutadas en plazo son peores que en el escenario de simulación anterior debido

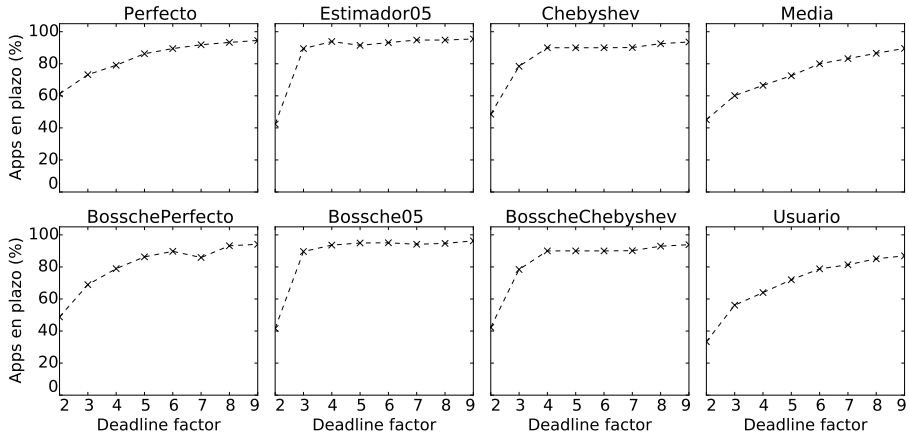



Figura 5.8: Carga Weibull, análisis límite temporal, tiempo de aprovisionamiento 100 segundos, porcentaje de aplicaciones ejecutadas en plazo

al error introducido en la planificación por el tiempo desconocido de aprovisionamiento de instancias de máquinas virtuales. Los planificadores Estimador05 y Bossche05 obtienen los mejores resultados. Aunque los resultados del algoritmo de planificación propuesto en este trabajo y el del trabajo de Van den Bossche obtienen resultados muy similares, debe tenerse en cuenta que el planificador de este trabajo obtiene mejores resultados en términos de aplicaciones ejecutadas en plazo para factores de límite temporal inferiores a 4 cuando se emplea el estimador perfecto. El análisis ANOVA para el porcentaje de aplicaciones ejecutadas en plazo indica que el factor de límite temporal (el tamaño del efecto es del 79 % - $\omega^2 = 0{,}793$) y el estimador del tiempo de procesamiento (el tamaño del efecto es del 11 % - $\omega^2 = 0{,}113$) tienen una influencia estadísticamente significativa en las diferencias de los resultados. El análisis post-hoc indica que existen diferencias estadísticamente significativas entre los valores de límite temporal 2 y 3 y el resto de valores, pero no se aprecian diferencias entre los estimadores, por lo que las diferencias entre los algoritmos al emplear el estimador Perfecto no son estadísticamente significativas.

La figura 5.9 muestra que todos los planificadores mandan a ejecución trabajos que terminan ejecutándose fuera de plazo. Esto se debe a que el tiempo de aprovisionamiento de las instancias de máquinas virtuales no se tienen en cuenta y por tanto el tiempo necesario para ejecutar un trabajo es subestima-

do. Los resultados de la simulación indican que en el caso de los planificadores basados en el estimador perfecto, todas los trabajos que son ejecutados fuera de plazo se corresponde con trabajos que son ejecutados en nuevas instancias recién aprovisionadas en el Cloud público. Debe notarse que los planificadores basados en los estimadores de Chebyshev y en el estimador basado en el trabajo de Van den Bossche obtienen mejores resultados que los planificadores basados en el estimador perfecto para factores de límite temporal entre 3 y 7 debido a que la sobreestimación de dichos estimadores compensa el error producido por los tiempos de aprovisionamiento desconocidos.

El análisis ANOVA para el porcentaje de aplicaciones ejecutadas fuera de plazo indica que el factor de límite temporal (el tamaño del efecto es del 47 % - $\omega^2 = 0{,}478$) y el estimador del tiempo de procesamiento (el tamaño del efecto es del 14 % - $\omega^2 = 0{,}143$) tienen una influencia estadísticamente significativa en las diferencias de los resultados. El análisis post-hoc indica que existen diferencias estadísticamente significativas entre el valor de límite temporal 2 y el resto de valores, y entre los estimadores Estimador05 y Media y entre el estimador de Chebyshev y el Estimador05. Estos resultados ponen de manifiesto que en este escenario el estimador de Media obtiene resultados que nos son distinguibles estadísticamente de los del estimador Perfecto, pero que si son diferenciables del resto de estimadores. Esto confirma la observación previa de que el estimador de Chebyshev y el Estimador05 obtienen resultados mejores debido a su sobreestimación.

Como se mostró en el escenario de simulación anterior, si la estimación es demasiado alta el número de aplicaciones inviables se incrementa, por lo que es necesario encontrar un balance entre la sobreestimación que compensa la información desconocida por el sistema y la subestimación que evite un alto porcentaje de aplicaciones inviables.

El análisis ANOVA para el tiempo medio de espera (figura 5.10) indica que el factor de límite temporal (el tamaño del efecto es del 55 % - $\omega^2 = 0{,}551$) y el estimador del tiempo de procesamiento (el tamaño del efecto es del 27 % - $\omega^2 = 0{,}271$) tienen una influencia estadísticamente significativa en las diferencias de los resultados. El análisis post-hoc indica que existen diferencias estadísticamente significativas entre los valores de límite temporal 7, 8 y 9 y el resto de valores, y entre los estimadores Estimador05 y Perfecto, los estimadores Estimador05 y Usuario, los estimadores Perfecto y de Chebyshev y los estimadores de Usuario y de Chebyshev. El efecto del estimador empleado sobre el tiempo medio de espera de las aplicaciones presenta el valor más alto de los escenarios analizados hasta el momento y pone de manifiesto que existen diferencias entre los estimadores Perfecto y Usuario y los estimadores de Chebyshev y de Usuario.

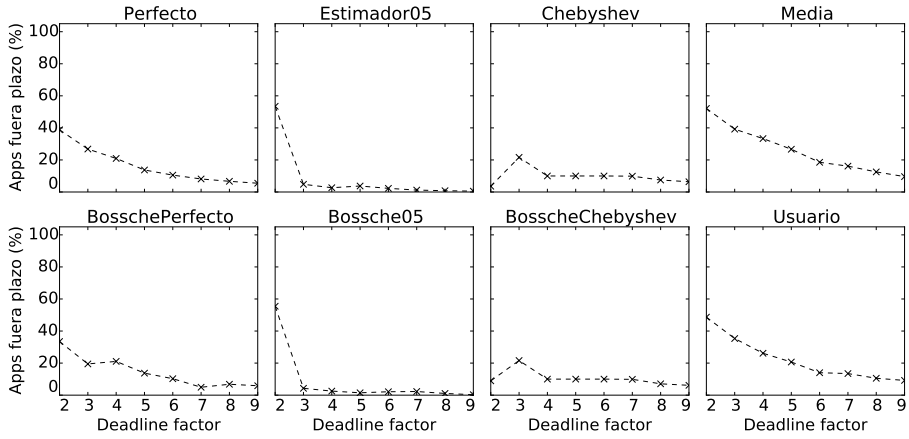



Figura 5.9: Carga Weibull, análisis límite temporal, tiempo de aprovisionamiento 100 segundos, porcentaje de aplicaciones fuera de plazo

Observando los datos, se aprecia como los tiempos medios de espera son más altos para el estimador Perfecto y de Usuario que para el resto.

Como se observa en la figura 5.11 el planificador propuesto en este trabajo obtiene mejores resultados que el planificador de Van den Bossche en términos de coste por aplicación ejecutada en plazo cuando se emplea tanto el estimador perfecto como el estimador basado en el trabajo de Van den Bossche. El análisis ANOVA realizado confirma que las diferencias entre ambos algoritmos son estadísticamente significativas, aunque con un tamaño del efecto limitado (tamaño del efecto del 11 % - $\omega^2 = 0{,}114$). El análisis ANOVA indica que el factor de límite temporal (el tamaño del efecto es del 34 % - $\omega^2 = 0{,}339$), el estimador de los tiempos de procesamiento (el tamaño del efecto es del 10 % - $\omega^2 = 0{,}107$) y el algoritmo de planificación (el tamaño del efecto es del 11 % - $\omega^2 = 0{,}114$) tienen una influencia estadísticamente significativa en las diferencias de los resultados. El análisis post-hoc indica, sin embargo, que no existen diferencias significativas entre los diferentes estimadores y que las diferencias en el factor de límite temporal se producen solo entre el valor 2 y el resto de valores. El análisis post-hoc si que confirma la existencia de diferencias entre los algoritmos de planificación.

El análisis ANOVA para el número de instancias de máquinas virtuales empleadas (figura 5.12) indica que el factor de límite temporal (el tamaño del

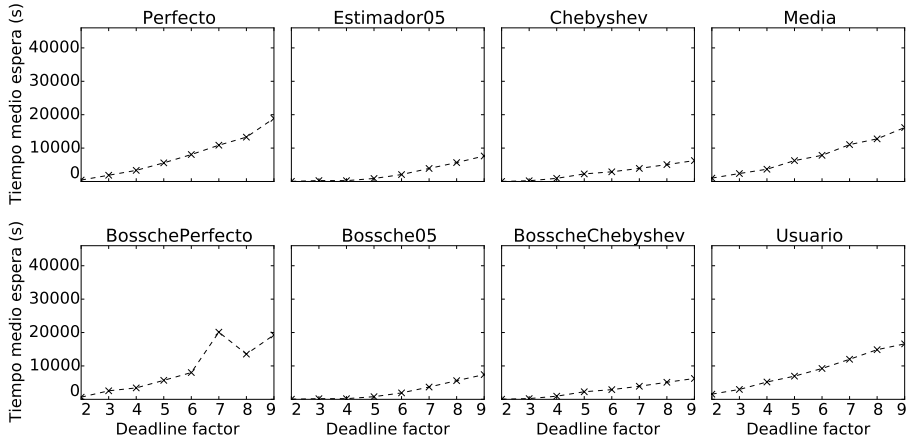



Figura 5.10: Carga Weibull, análisis límite temporal, tiempo de aprovisionamiento 100 segundos, tiempo medio de espera por aplicación

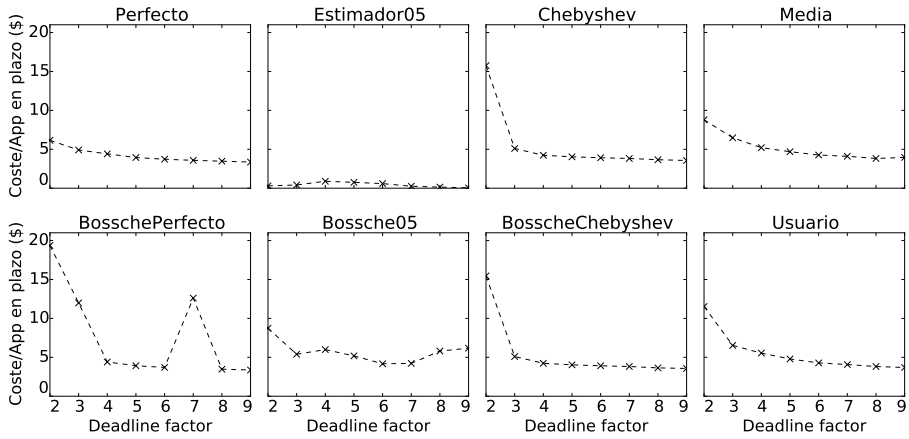



Figura 5.11: Carga Weibull, análisis límite temporal, tiempo de aprovisionamiento 100 segundos, coste por aplicación ejecutada en plazo

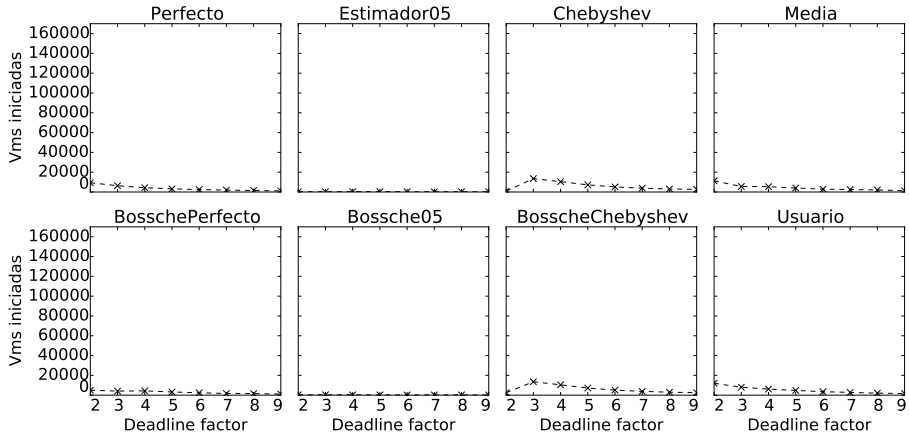



Figura 5.12: Carga Weibull, análisis límite temporal, tiempo de aprovisionamiento 100 segundos, instancias de máquinas virtuales iniciadas

efecto es del 21 % - $\omega^2 = 0{,}212$) y el estimador del tiempo de procesamiento (el tamaño del efecto es del 33 % - $\omega^2 = 0{,}334$) tienen una influencia estadísticamente significativa en las diferencias de los resultados. El análisis post-hoc indica que existen diferencias estadísticamente significativas entre los valores de límite temporal 3 y 9 y entre el estimador Estimador05 y el resto de estimadores.

### 5.2.3. Carga Weibull para cualquier tiempo de aprovisionamiento

En esta sección se analizan los resultados de todas las simulaciones que emplean la carga basada en una distribución Weibull considerando cualquier tiempo de aprovisionamiento de nuevas instancias de máquinas virtuales. Para realizar este análisis se han unido los resultados numéricos presentados en las tablas B.1 y B.2 del anexo B.

La tabla 5.4 muestra los tamaños de los efectos de las variables independientes sobre las diferentes variables dependientes para aquellos casos en los que la variable independiente es estadísticamente significativa y cuando el análisis post-hoc posterior es capaz de encontrar niveles con diferencias. El análisis ANOVA muestra que, como era de esperar, el factor del límite temporal de ejecución explica las diferencias en los valores de todas las variables independientes con un

| | Factor | Aprovisionamiento | Estimador | Algoritmo |
|---|---|---|---|---|
| % en plazo | 0.558 | 0.121 | 0.085 | |
| % fuera plazo | 0.328 | 0.117 | 0.077 | |
| Tiempo de espera | 0.507 | | 0.187 | |
| Coste | 0.286 | 0.040 | | 0.051 |
| Núm. de instancias | 0.305 | | 0.233 | |

Tabla 5.4: Análisis ANOVA para la carga Weibull para cualquier tiempo de aprovisionamiento - Tamaño del efecto

tamaño de efecto cercano al 30 % en todos los casos, y en algunos casos, como el porcentaje de aplicaciones ejecutadas en plazo y el tiempo de espera, con un tamaño de efecto superior al 50 %. Como segundo factor (variable independiente) con mayor efecto sobre las variables dependientes se encuentra el tiempo de aprovisionamiento de nuevas instancias de máquinas virtuales, aunque este factor no es estadísticamente significativo para todas las variables dependientes. El tercer factor en importancia es el estimador empleado, aunque su efecto se encuentra por debajo del 10 % para el porcentaje de aplicaciones ejecutadas en plazo y fuera de plazo, su efecto sí es relevante sobre el tiempo de espera y sobre el número de instancias de máquinas virtuales empleadas. Finalmente, cabe destacar que el algoritmo empleado solo tiene impacto en el coste medio por cada aplicación ejecutada en plazo, aunque con un tamaño de efecto del 5 %.

Estos resultados ponen de manifiesto que si bien el uso de uno u otro estimador tiene un impacto estadísticamente significativo sobre las características de la planificación, los dos algoritmos analizados no tienen un impacto significativo.

### 5.2.4. Carga Normal con tiempo de aprovisionamiento ideal

Las figuras 5.13, 5.14, 5.15, 5.16 y 5.17 muestran los resultados de la simulación con la carga de trabajo modelada mediante una distribución Normal y con tiempos de aprovisionamiento de las instancias de máquinas virtuales ideales ($avt_{cti} - rqt_{cti} = 0$). La tabla B.3 del anexo B contiene los resultados numéricos.

Como se muestra en la figura 5.13 los planificadores basados en el estimador perfecto obtienen los mejores resultados, aunque los que están basados en el estimador de Chebyshev obtienen resultados muy similares para factores de límite temporal mayores de 2.

Si se analizan las figuras 5.13 y 5.14 se puede observar que para los planificadores basados en la desigualdad de Chebyshev el número de aplicaciones

ejecutadas en plazo es cero y el número de aplicaciones ejecutadas fuera de plazo es también cero. Esto quiere decir que el 100 % de las aplicaciones son marcadas como inviables por el planificador y muestra el efecto de la sobreestimación de una manera mucho más clara que en el caso de la carga de trabajo basada en una distribución Weibull. Esto se debe a las características de la carga de trabajo basada en una distribución Normal y al método de estimación empleado por el estimador de Chebyshev. La desviación estándar de los tiempos de procesamiento base para cada usuario en la carga basada en una distribución Normal es 0,5 veces la media de los tiempos de procesamiento mientras que el estimador basado en la desigualdad de Chebyshev produce estimaciones de acuerdo a la ecuación 5.14. Si se asume que $s_{uct} = 0,5 \cdot m_{uct}$ la ecuación puede ser reescrita según se muestra en la ecuación 5.4, lo que significa que para esta carga de trabajo el estimador de Chebyshev producirá estimaciones que tenderán a ser dos veces el tiempo medio de procesamiento y que coincide con el valor del límite temporal cuando el factor es 2. Este escenario de simulación muestra que la sobreestimación con límites temporales de ejecución ajustados es problemática para los planificadores cuando se emplea el estimador de Chebyshev. El estimador de media no se ve tan afectado por este problema aunque en general obtiene peores resultados.

$$EC(u, a, j, c, t, i, k) = m_{uct} + 2s_{uct} = m_{uct} + 2 \cdot (0,5 \cdot m_{uct}) = 2 \cdot m_{uct} \quad (5.4)$$

El análisis ANOVA para el porcentaje de aplicaciones ejecutadas en plazo (figura 5.13) indica que el factor de límite temporal (el tamaño del efecto es del 53 % - $\omega^2 = 0,539$) y el estimador del tiempo de procesamiento (el tamaño del efecto es del 7 % - $\omega^2 = 0,075$) tienen una influencia estadísticamente significativa en las diferencias de los resultados. El análisis post-hoc indica que existen diferencias estadísticamente significativas entre el valor de límite temporal 2 y el resto de valores, pero no se encuentran diferencias significativas entre los estimadores.

En la figura 5.14 se puede apreciar que el porcentaje de aplicaciones ejecutadas fuera de plazo obtenidas por los estimadores Perfecto y de Chebyshev es inferior al del resto de estimadores independientemente del algoritmo de planificación empleado. El análisis ANOVA confirma que existen diferencias estadísticamente significativas que se explican por el factor de límite temporal (el tamaño del efecto es del 26 % - $\omega^2 = 0,260$) y por el estimador empleado (el tamaño del efecto es del 24 % - $\omega^2 = 0,245$) con un tamaño de efecto muy similar. El análisis post-hoc posterior confirma que los estimadores Perfecto y

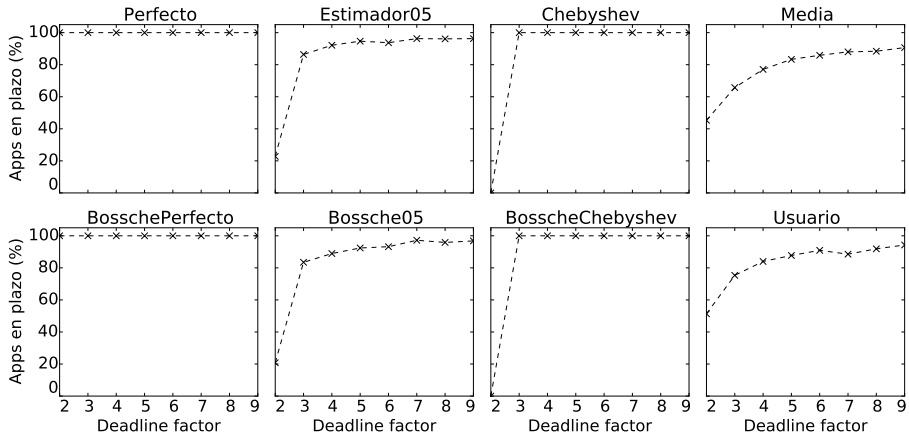



Figura 5.13: Carga normal, análisis límite temporal, tiempo de aprovisionamiento 0 segundos, porcentaje de aplicaciones ejecutadas en plazo

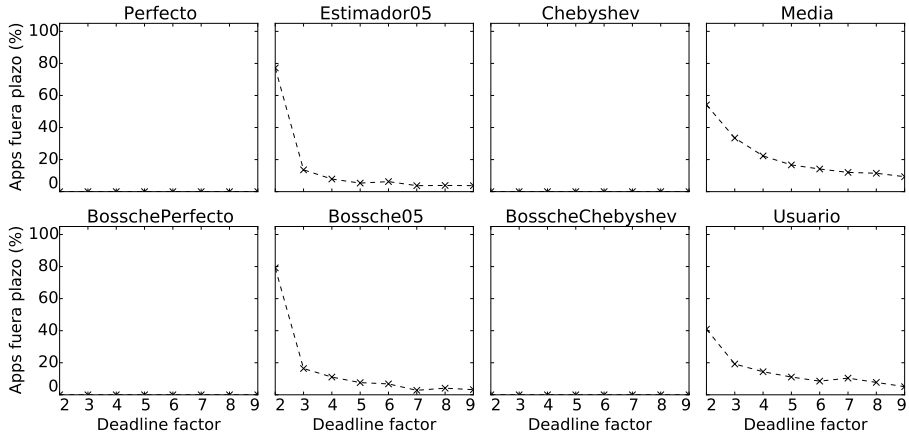



Figura 5.14: Carga normal, análisis límite temporal, tiempo de aprovisionamiento 0 segundos, porcentaje de aplicaciones fuera de plazo

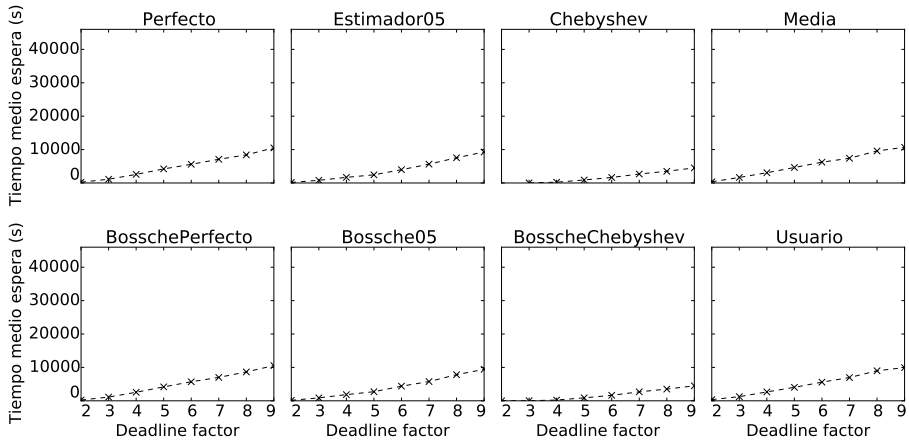



Figura 5.15: Carga normal, análisis límite temporal, tiempo de aprovisionamiento 0 segundos, tiempo medio de espera por aplicación

de Chebyshev obtienen diferencias estadísticamente significativas con respecto a los estimadores de media y el Estimador05.

El análisis ANOVA para el tiempo medio de espera (figura 5.15) indica que el factor de límite temporal (el tamaño del efecto es del 75 % - $\omega^2 = 0{,}757$) y el estimador del tiempo de procesamiento (el tamaño del efecto es del 17 % - $\omega^2 = 0{,}171$) tienen una influencia estadísticamente significativa en las diferencias de los resultados. El análisis post-hoc indica que existen diferencias estadísticamente significativas entre los valores de límite temporal y entre el estimador de Chebyshev y los estimadores de media y Perfecto.

En el caso del coste medio por aplicación ejecutada en plazo el análisis ANOVA indica que solo el factor de límite temporal tiene una influencia estadísticamente significativa en las diferencias de los resultados con un tamaño de efecto del 14.8 %. El análisis post-hoc solo encuentra diferencias significativas entre el valor 2 y los valores 8 y 9.

Finalmente, el análisis ANOVA para el número de instancias de máquinas virtuales iniciadas (figura 5.17) indica que el factor de límite temporal (el tamaño del efecto es del 26 % - $\omega^2 = 0{,}265$) y el estimador del tiempo de procesamiento (el tamaño del efecto es del 14 % - $\omega^2 = 0{,}140$) tienen una influencia estadísticamente significativa en las diferencias de los resultados. El análisis post-hoc indica que existen diferencias estadísticamente significativas entre el valor de

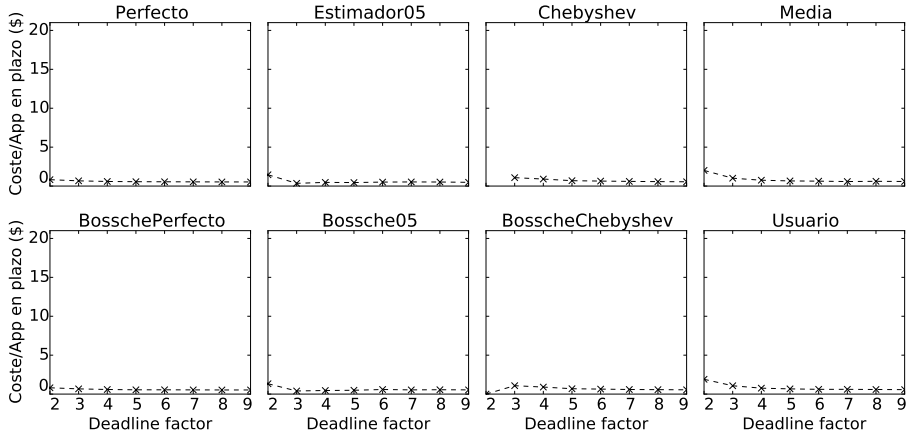



Figura 5.16: Carga normal, análisis límite temporal, tiempo de aprovisionamiento 0 segundos, coste por aplicación ejecutada en plazo

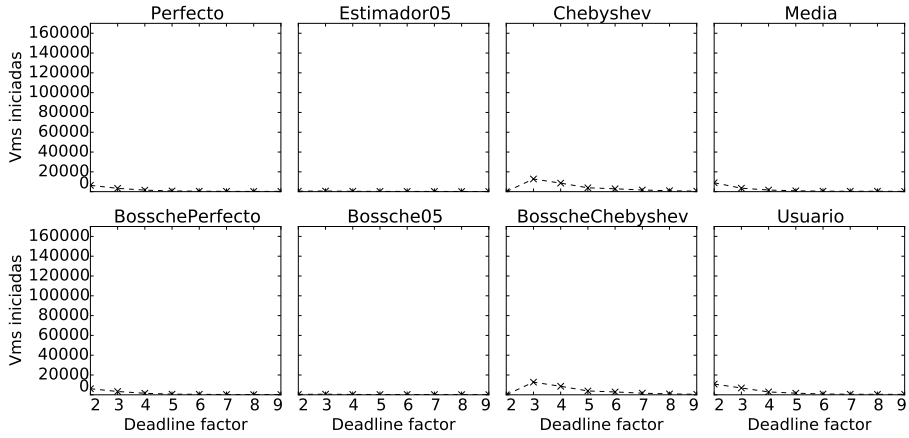



Figura 5.17: Carga normal, análisis límite temporal, tiempo de aprovisionamiento 0 segundos, instancias de máquinas virtuales iniciadas

límite temporal 3 y los valores 6, 7, 8 y 9 y entre el estimador Estimador05 y el estimador de Chebyshev.

### 5.2.5. Carga Normal con tiempo de aprovisionamiento de 100 segundos

Las figuras 5.18, 5.19, 5.20, 5.21 y 5.22 muestran los resultados de la simulación con la carga de trabajo modelada mediante una distribución Normal y con tiempos de aprovisionamiento de las instancias de máquinas virtuales de 100 segundos ($avt_{cti} - rqt_{cti} = 100$). La tabla B.4 del anexo B contiene los resultados numéricos.

Los resultados en este escenario de simulación son similares a los escenario anterior con tiempos de aprovisionamiento ideales, pero los planificadores basados en el estimador perfecto no llegan a obtener un 100 % de aplicaciones ejecutadas en plazo cuando el factor de límite de ejecución es dos porque los algoritmos no tienen en cuenta el tiempo de aprovisionamiento. (ver figuras 5.18 y 5.19).

Con esta carga de trabajo los planificadores basados en los estimadores perfecto y de Chebyshev obtienen un 100 % de aplicaciones ejecutadas en plazo para factores de límite temporal mayores que 3. Cuando el factor de límite temporal es 2 los planificadores basados en el trabajo de Van den Bossche obtienen resultados ligeramente superiores en términos de aplicaciones ejecutadas en plazo, aunque con factores de límite temporal superiores estas diferencias desparecen. Por lo general en este escenario de simulación todas las combinaciones de algoritmo de planificación y estimador de los tiempos de procesamiento tienen un comportamiento muy similar. El análisis ANOVA para el porcentaje de aplicaciones ejecutadas en plazo confirma esta observación, dado que solo el factor de límite temporal con un tamaño de efecto del 81 % y el estimador con un tamaño de efecto del 5 % tienen una influencia estadísticamente significativa en las diferencias de los resultados. Además, el análisis post-hoc indica que solo existen diferencias significativas entre el valor del límite temporal 2 y el resto de valores y no se encuentran diferencias entre los estimadores. Esto indica que desde el punto de vista estadístico ni el estimador ni el algoritmo de planificación son factores significativos a la hora de explicar el porcentaje de aplicaciones ejecutadas en plazo.

Mientras que en los escenarios de simulación basados en la carga de trabajo Weibull los planificadores que emplean el estimador de media obtiene resultados mejores o iguales que los planificadores que usan el estimador de usuario, con la carga basada en una distribución Normal el estimador de usuario obtiene mejores

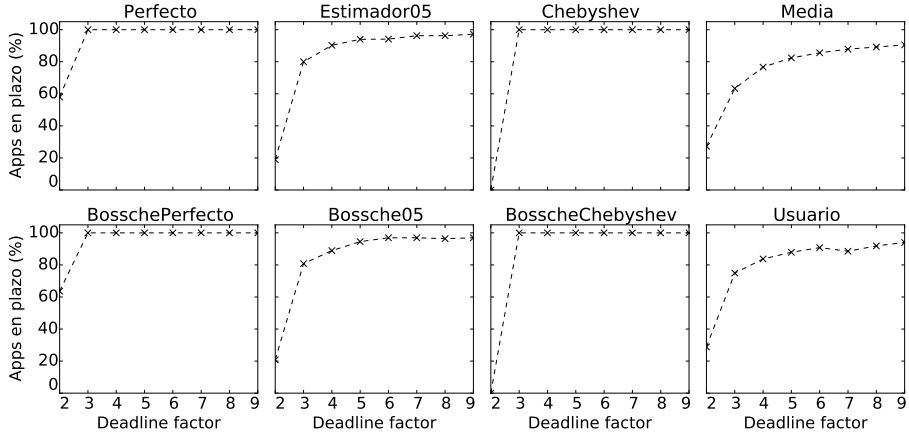



Figura 5.18: Carga normal, análisis límite temporal, tiempo de aprovisionamiento 100 segundos, porcentaje de aplicaciones ejecutadas en plazo

resultados en términos de aplicaciones ejecutadas en plazo. Por el contrario el estimador de Chebyshev obtiene resultados similares a los de estimador perfecto con los dos tipos de cargas de trabajo porque la desigualdad de Chebyshev no depende en la distribución de probabilidad de los tiempos de procesamiento y puede ser aplicada a cualquier distribución de probabilidad para la que sea posible obtener la media y varianza muestral.

Desde el punto de vista de las aplicaciones ejecutadas fuera de plazo (figura 5.19) se observan diferencias entre los algoritmos que emplean el estimador basado en la desigualdad de Chebyshev y el resto de estimadores. Debido a la sobreestimación del estimador de Chebyshev este estimador mantiene siempre a 0 el porcentaje de aplicaciones que se ejecutan fuera de plazo, lo que significa que muchas aplicaciones son marcadas como inviables cuando el factor de límite temporal es 2. El análisis ANOVA confirma estas observaciones e indica que el factor de límite temporal (el tamaño del efecto es del 52 % - $\omega^2 = 0{,}522$) y el estimador del tiempo de procesamiento (el tamaño del efecto es del 16 % - $\omega^2 = 0{,}168$) tienen una influencia estadísticamente significativa en las diferencias de los resultados. El análisis post-hoc indica que existen diferencias estadísticamente significativas entre el valor de límite temporal 2 y el resto de valores y entre el estimador de media y el estimador de Chebyshev.

Los tiempos de espera medios mostrados en la figura 5.20 muestran valores

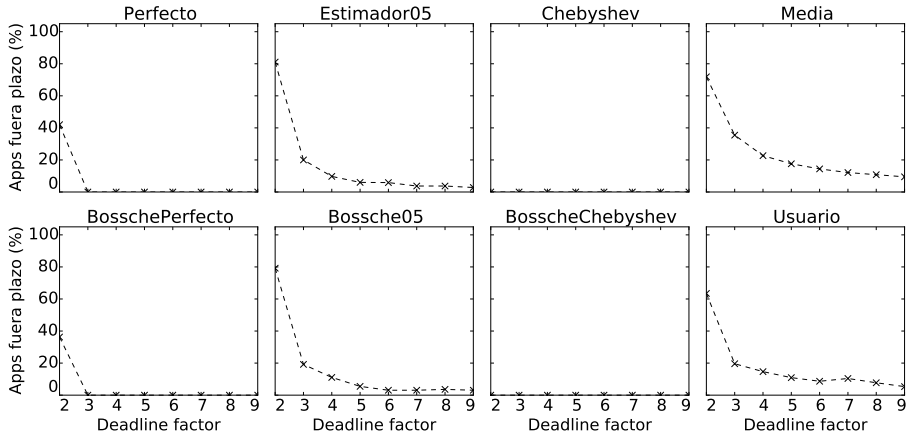



Figura 5.19: Carga normal, análisis límite temporal, tiempo de aprovisionamiento 100 segundos, porcentaje de aplicaciones fuera de plazo

similares para todos los algoritmos y estimadores, salvo quizás en el caso del estimador de Chebyshev que tiene tiempos de espera ligeramente inferiores debido a la sobreestimación que introduce. El análisis ANOVA indica que el factor de límite temporal (el tamaño del efecto es del 75 % - $\omega^2 = 0{,}758$) y el estimador del tiempo de procesamiento (el tamaño del efecto es del 17 % - $\omega^2 = 0{,}170$) tienen una influencia estadísticamente significativa en las diferencias de los resultados. El análisis post-hoc indica que existen diferencias estadísticamente significativas entre los valores del factor de límite temporal y entre el estimador Perfecto y el estimador de Chebyshev.

Como se puede observar en la figura 5.21 el coste medio por aplicación ejecutada en plazo se mantiene muy bajo en todos los algoritmos y estimadores sin apreciarse diferencias relevantes, salvo quizás cuando el factor de límite temporal toma el valor 2. El análisis ANOVA del coste medio por aplicación confirma estas observaciones dado que solo el factor de límite temporal tiene una influencia estadísticamente significativa sobre las diferencias en el coste (el tamaño del efecto es del 33 %). El análisis post-hoc confirma las diferencias entre el valor del factor de límite temporal 2 y el resto de valores.

El número de instancias de máquinas virtuales empleadas se mantiene muy bajo para todos los algoritmos y estimadores, salvo en el caso del estimador de Chebyshev debido a su sobreestimación como se muestra en la figura 5.22.

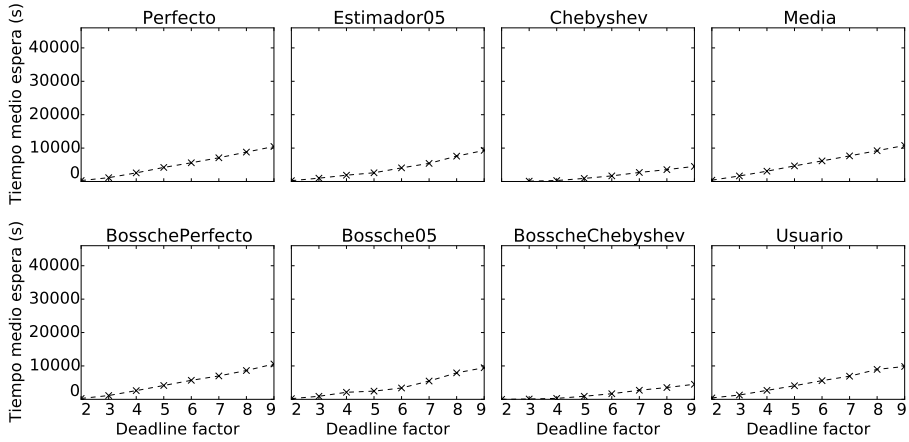



Figura 5.20: Carga normal, análisis límite temporal, tiempo de aprovisionamiento 100 segundos, tiempo medio de espera por aplicación

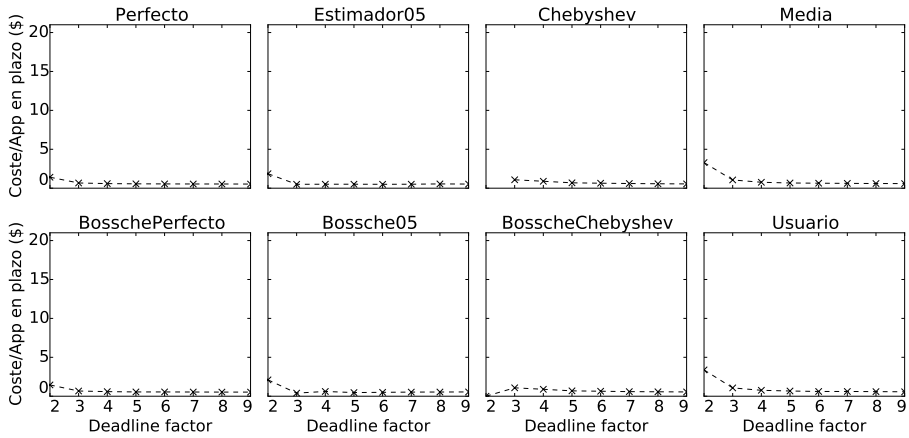



Figura 5.21: Carga normal, análisis límite temporal, tiempo de aprovisionamiento 100 segundos, coste por aplicación ejecutada en plazo

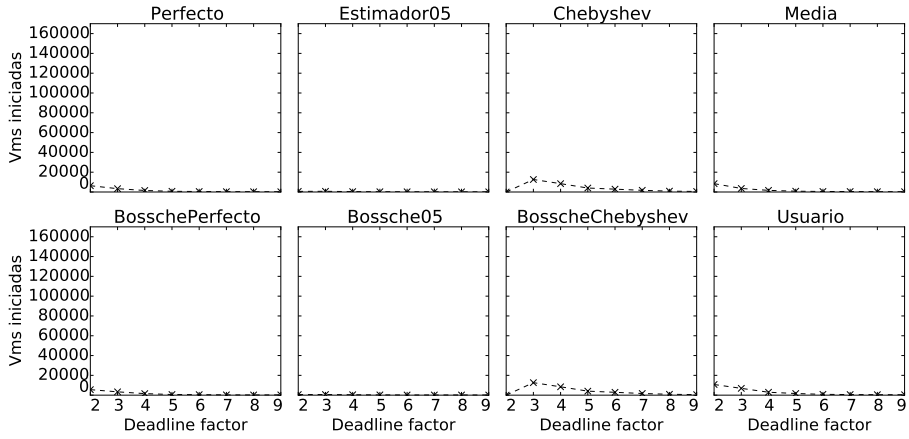



Figura 5.22: Carga normal, análisis límite temporal, tiempo de aprovisionamiento 100 segundos, instancias de máquinas virtuales iniciadas

Cabe destacar que el mayor número de instancias usadas no se corresponde directamente con un mayor coste medio por aplicación ejecutada en plazo, ya que influye también el tiempo de uso de cada instancia. El análisis ANOVA indica que el factor de límite temporal (el tamaño del efecto es del 27 % - $\omega^2 =$ 0,270) y el estimador del tiempo de procesamiento (el tamaño del efecto es del 14 % - $\omega^2 = 0{,}145$) tienen una influencia estadísticamente significativa en las diferencias de los resultados. El análisis post-hoc indica que existen diferencias estadísticamente significativas entre el valor del factor de límite temporal 3 y los valores 6, 7, 8 y 9 y entre el estimador Estimador05 y el estimador de Chebyshev.

### 5.2.6. Carga Normal para cualquier tiempo de aprovisionamiento

En esta sección se analizan los resultados de todas las simulaciones que emplean la carga basada en una distribución Normal para cualquier tiempo de aprovisionamiento de nuevas instancias de máquinas virtuales. Para realizar este análisis se han unido los resultados numéricos presentados en las tablas B.3 y B.4 del anexo B.

La tabla 5.5 muestra los tamaños de los efectos de las variables independientes sobre las diferentes variables dependientes para aquellos casos en los que

| | Factor | Aprovisionamiento | Estimador | Algoritmo |
|---|---|---|---|---|
| % en plazo | 0.691 | | 0.068 | |
| % fuera plazo | 0.413 | | 0.210 | |
| Tiempo de espera | 0.761 | | 0.173 | |
| Coste | 0.273 | | | |
| Núm. de instancias | 0.305 | | 0.164 | |

Tabla 5.5: Análisis ANOVA para la carga Normal para cualquier tiempo de aprovisionamiento - Tamaño del efecto

la variable independiente es estadísticamente significativa y cuando el análisis post-hoc posterior es capaz de encontrar niveles con diferencias.

El análisis ANOVA muestra que, como era de esperar, el factor del límite temporal de ejecución explica las diferencias en los valores de todas las variables independientes con un tamaño de efecto cercano al 30 % en todos los casos, y en algunos casos como el porcentaje de aplicaciones ejecutadas en plazo y el tiempo de espera con un tamaño de efecto superior al 50 %. Como segundo factor (variable independiente) sobre las variables dependientes se encuentra el tiempo de aprovisionamento de nuevas instancias de máquinas virtuales, aunque este factor no es estadísticamente significativo para todas las variables dependientes. Al contrario de lo que ocurría con la carga Weibull, el tiempo de aprovisionamiento no es estadísticamente significativo para ninguna de las variables dependientes, poniendo de manifiesto el efecto del tipo de carga sobre los resultados. El algoritmo de planificación tampoco resulta significativo sobre ninguna variable.

Estos resultados confirman la observación de que si bien el uso de uno u otro estimador tiene un impacto estadísticamente significativo sobre las características de la planificación, los dos algoritmos analizados no lo tienen.

### 5.2.7. Carga de Google con tiempo de aprovisionamiento ideal

Las figuras 5.23, 5.24, 5.25, 5.26 y 5.27 muestran los resultados de la simulación con la carga de trabajo de Google y con tiempos de aprovisionamiento de las instancias de máquinas virtuales ideales ($avt_{cti} - rqt_{cti} = 0$). La tabla B.5 del anexo B contiene los resultados numéricos.

La figura 5.23 muestra que los mejores resultados en términos de aplicaciones ejecutadas en plazo tras los planificadores que emplean el estimador perfecto se obtienen con el planificador propuesto en este trabajo y el estimador de media.

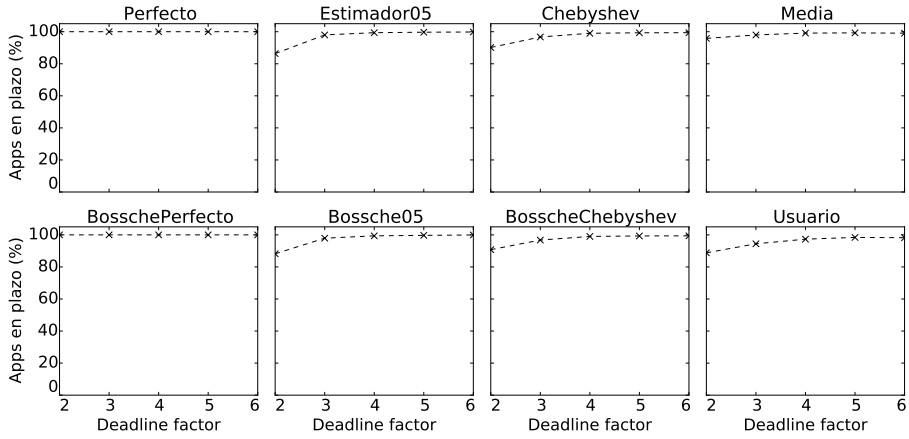



Figura 5.23: Carga Google, análisis límite temporal, tiempo de aprovisionamiento 0 segundos, porcentaje de aplicaciones ejecutadas en plazo

Aunque el planificador basado en el estimador de usuario obtiene en general peores resultados es el escenario de simulación en el que más aproxima al resto de estimadores, superando incluso en términos de aplicaciones ejecutadas fuera de plazo (figura 5.24) al estimador basado en el trabajo de Van den Bossche. El análisis ANOVA para el porcentaje de aplicaciones ejecutadas en plazo indica que el factor de límite temporal (el tamaño del efecto es del 50 % - $\omega^2 = 0{,}507$) y el estimador del tiempo de procesamiento (el tamaño del efecto es del 13 % - $\omega^2 = 0{,}136$) tienen una influencia estadísticamente significativa en las diferencias de los resultados. El análisis post-hoc indica que existen diferencias estadísticamente significativas entre el valor del factor de límite temporal 2 y el resto de valores, pero no se encuentran diferencias significativas entre los diferentes estimadores.

Con esta carga de trabajo los planificadores basados en el estimador perfecto consiguen ejecutar un 0 % de aplicaciones fuera de plazo y el resto de estimadores menos de un 1 % cuando el factor de límite temporal es mayor de 4 tal y como se muestra en la figura 5.24. El análisis ANOVA para el porcentaje de aplicaciones ejecutadas fuera de plazo indica que el factor de límite temporal (el tamaño del efecto es del 14 % - $\omega^2 = 0{,}147$) y el estimador del tiempo de procesamiento (el tamaño del efecto es del 15 % - $\omega^2 = 0{,}151$) tienen una influencia estadísticamente significativa en las diferencias de los resultados. Sin embargo,

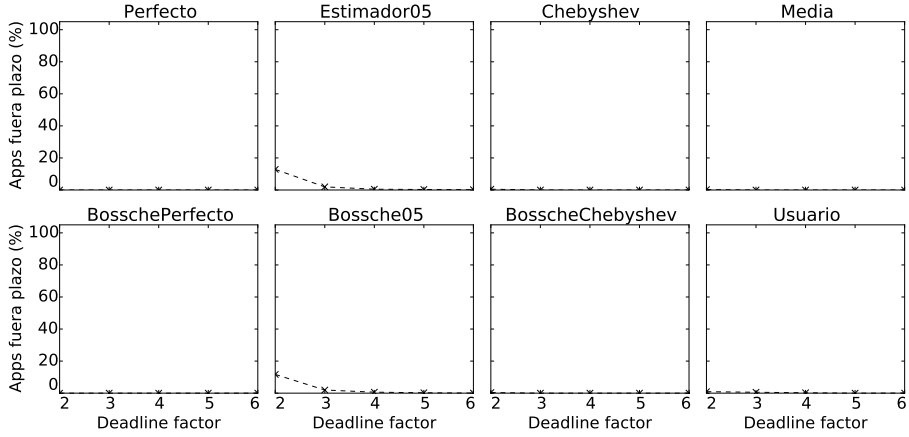



Figura 5.24: Carga Google, análisis límite temporal, tiempo de aprovisionamiento 0 segundos, porcentaje de aplicaciones fuera de plazo

el análisis post-hoc no encuentra diferencias significativas entre los valores de ninguno de los dos factores.

El análisis ANOVA para los resultados del tiempo medio de espera (figura 5.25) indica que el factor de límite temporal (el tamaño del efecto es del 77 % - $\omega^2 = 0{,}777$) y el estimador del tiempo de procesamiento (el tamaño del efecto es del 18 % - $\omega^2 = 0{,}185$) tienen una influencia estadísticamente significativa en las diferencias de los resultados. El análisis post-hoc indica que existen diferencias estadísticamente significativas entre los valores del factor de límite temporal 2 y 3 y el resto de valores, pero no se encuentran diferencias significativas entre los diferentes estimadores.

Aunque todas las combinaciones de planificador y estimador presentan resultados similares en términos de aplicaciones ejecutadas en plazo, el coste por aplicación ejecutada en plazo sí que presenta grandes diferencias como se muestra en la figura 5.26. Estas diferencias se explican por los diferentes grados de sobreestimación de los estimadores, cuanto menor sobreestimación, menor coste. El análisis ANOVA para el coste indica que el factor de límite temporal (el tamaño del efecto es del 51 % - $\omega^2 = 0{,}517$) y el estimador del tiempo de procesamiento (el tamaño del efecto es del 17 % - $\omega^2 = 0{,}177$) tienen una influencia estadísticamente significativa en las diferencias de los resultados. El análisis post-hoc indica que existen diferencias estadísticamente significativas entre el

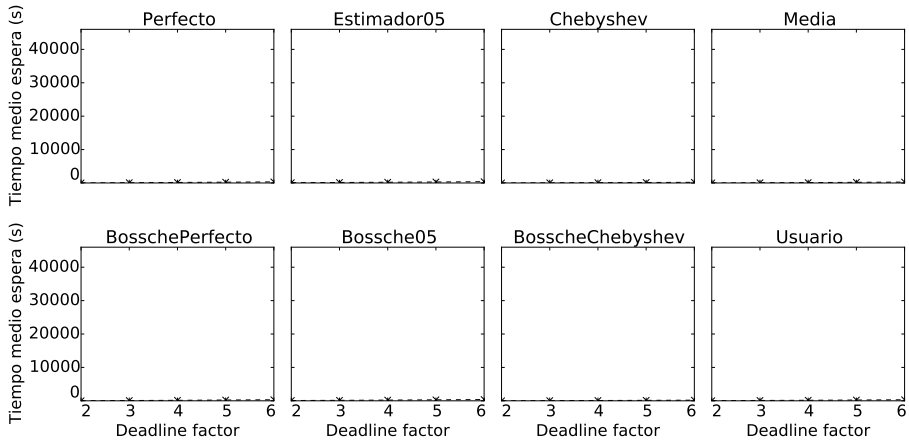



Figura 5.25: Carga Google, análisis límite temporal, tiempo de aprovisionamiento 0 segundos, tiempo medio de espera por aplicación

valor del factor de límite temporal 2 y el resto de valores, pero no se encuentran diferencias significativas entre los diferentes estimadores.

El análisis ANOVA para el número de instancias de máquinas virtuales iniciadas indica que el factor de límite temporal (el tamaño del efecto es del 74 % - $\omega^2 = 0{,}743$) y el estimador del tiempo de procesamiento (el tamaño del efecto es del 13 % - $\omega^2 = 0{,}136$) tienen una influencia estadísticamente significativa en las diferencias de los resultados. El análisis post-hoc indica que existen diferencias estadísticamente significativas entre el valor del factor de límite temporal 2 y el resto de valores, pero no se encuentran diferencias significativas entre los diferentes estimadores.

En este escenario de simulación destaca el hecho de que el análisis ANOVA considera estadísticamente significativa la influencia del estimador sobre las variables dependientes en un primer momento, aunque en ninguno de los análisis post-hoc se encuentran diferencias significativas entre los diferentes niveles de dicho factor. Se pone de manifiesto por tanto que todos los estimadores y los algoritmos de planificación tienen un comportamiento muy similar, cosa que no ocurría con las cargas basadas en una distribución Weibull y una distribución Normal.

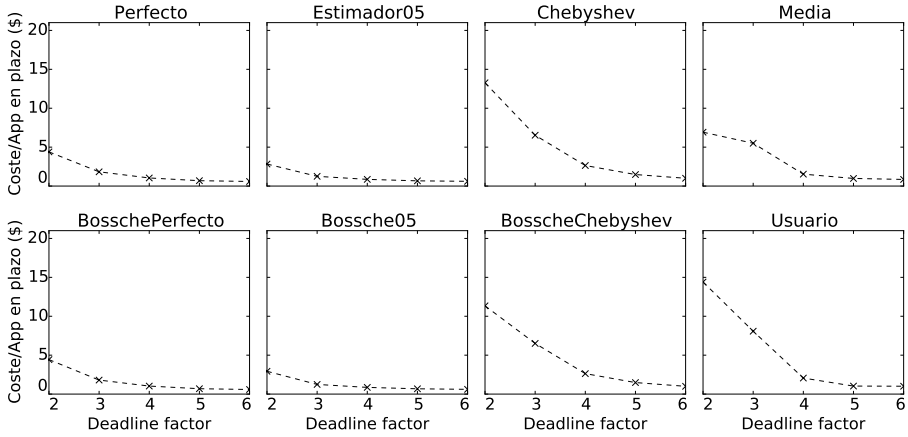



Figura 5.26: Carga Google, análisis límite temporal, tiempo de aprovisionamiento 0 segundos, coste por aplicación ejecutada en plazo

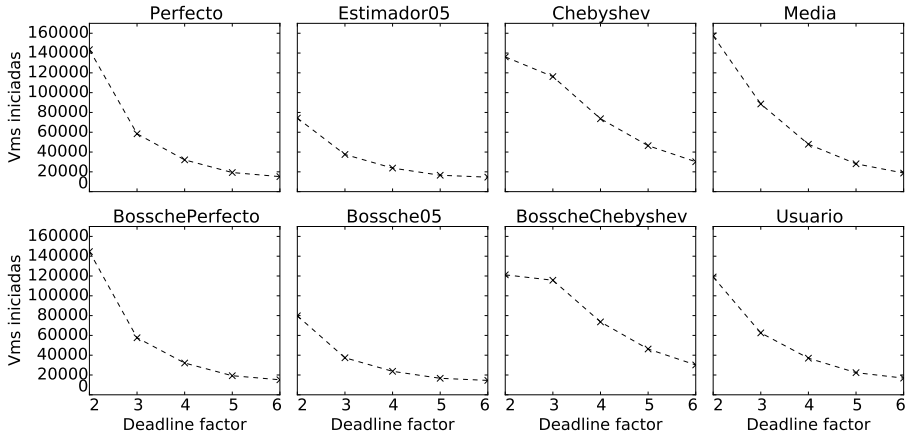



Figura 5.27: Carga Google, análisis límite temporal, tiempo de aprovisionamiento 0 segundos, instancias de máquinas virtuales iniciadas

| | Carga | Factor | Aprovisionamiento | Estimador | Algoritmo |
|---|---|---|---|---|---|
| % en plazo | 0.036 | 0.519 | 0.025 | 0.052 | |
| % fuera plazo | 0.035 | 0.316 | 0.031 | 0.112 | |
| Tiempo de espera | 0.070 | 0.454 | | 0.117 | |
| Coste | 0.365 | 0.157 | | | |
| Núm. de instancias | 0.506 | 0.084 | | | |

Tabla 5.6: Análisis ANOVA para el análisis de límite temporal independientemente de la carga y del tiempo de aprovisionamiento - Tamaño del efecto

### 5.2.8. Análisis independiente de la carga y el tiempo de aprovisionamiento

En esta sección se presenta el análisis ANOVA realizado sobre los resultados de todos los escenarios de simulación relativos al análisis por límite temporal, independientemente de la carga de trabajo y del tiempo de aprovisionamiento de nuevas instancias de máquinas virtuales. Estos dos factores se convierten en variables independientes del modelo ANOVA empleando la ecuación 5.5. Para realizar este análisis se han unido los resultados numéricos presentados en las tablas B.1, B.2, B.3, B.4 y B.5 del anexo B.

$$variable \sim C(carga) + C(aprovisionamiento) + C(factor) + C(algoritmo) + C(estimador) \quad (5.5)$$

La tabla 5.6 muestra los tamaños de los efectos de las variables independientes sobre las diferentes variables dependientes para aquellos casos en los que la variable independiente es estadísticamente significativa y cuando el análisis post-hoc posterior es capaz de encontrar niveles con diferencias.

Como se puede observar el factor con mayor impacto sobre los resultados es el factor de límite temporal, seguido de la carga de trabajo, el estimador y finalmente el tiempo de aprovisionamiento, aunque estos dos últimos factores tienen tamaños de efecto muy pequeños. La carga de trabajo solo alcanza tamaños de efecto significativos sobre el coste medio por cada aplicación ejecutada en plazo y sobre el número de instancias de máquinas virtuales empleadas. En todos los análisis post-hoc la carga de Google obtiene diferencias estadísticamente significativas con respecto a la cargas Weibull y Normal. En el caso del tiempo medio de espera y del coste medio por aplicación también existen diferencias

significativas entre las cargas Normal y Weibull. Esto pone de manifiesto el diferente comportamiento de los planificadores en función de la carga de trabajo de entrada y aumenta el valor de la validación llevada a cabo en este trabajo con diferentes cargas.

El nulo impacto del algoritmo de planificación refuerza la idea de que el algoritmo presentado en este trabajo obtiene resultados muy similares a los de los algoritmos ya existentes en el estado del arte. Por contra, el uso de los estimadores del tiempo de procesamiento sí que tiene impacto en los resultados.

## 5.3. Análisis por ratio de llegada de trabajos

En las simulaciones de esta sección el factor de límite temporal de ejecución se fija a 4 y la tasa de llegada de aplicaciones se varia desde Poisson(0,002) a Poisson(8).

### 5.3.1. Carga Weibull con tiempo de aprovisionamiento ideal

Las figuras 5.28, 5.29, 5.30, 5.31 y 5.32 muestran los resultados de la simulación empleando la carga de trabajo basada en la distribución Weibull y con tiempos de aprovisionamiento de instancias de máquinas virtuales ideales, es decir, de 0 segundos. La tabla B.6 del anexo B contiene los resultados numéricos.

En esta simulación los planificadores que emplean el estimador perfecto (Perfecto y BosschePerfecto) obtienen un 100 % de aplicaciones ejecutadas en plazo como se puede ver en la figura 5.28 (como era de esperar en un sistema con máquinas virtuales ideales y con un número ilimitado de instancias). Los planificadores que emplean el estimador de Chebyshev (Chebyshev y Bossche-Chebyshev) obtienen resultados muy similares y mejores que el resto de las combinaciones de planificador y estimador. El análisis ANOVA para el porcentaje de aplicaciones ejecutadas en plazo indica que solo el estimador del tiempo de procesamiento (el tamaño del efecto es del 72 % - $\omega^2 = 0{,}722$) tiene una influencia estadísticamente significativa en las diferencias de los resultados. El análisis post-hoc indica que existen diferencias estadísticamente significativas entre el valor del Estimador05 y el resto de estimadores lo que confirma lo observado en las gráficas de los resultados.

El análisis ANOVA para el porcentaje de aplicaciones ejecutadas fuera de plazo (figura 5.29) indica que solo el estimador del tiempo de procesamiento (el tamaño del efecto es del 73 % - $\omega^2 = 0{,}739$) tiene una influencia estadísticamente

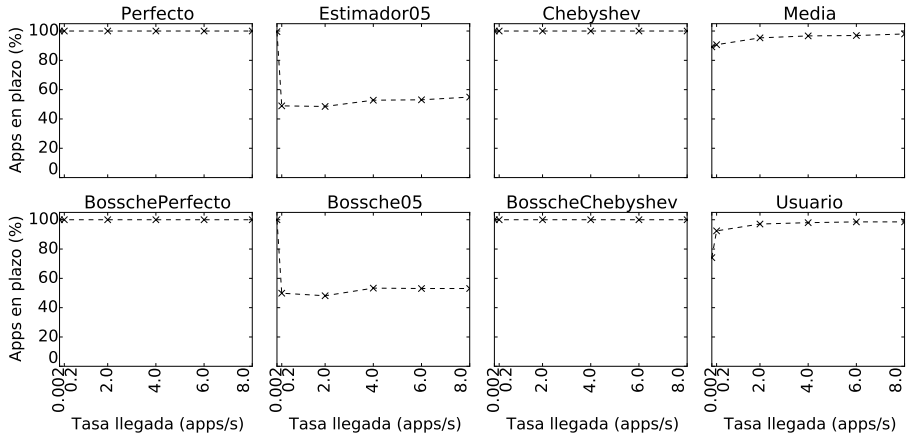



Figura 5.28: Carga Weibull, análisis por ratio de llegada, tiempo de aprovisionamiento 0 segundos, porcentaje de aplicaciones ejecutadas en plazo

significativa en las diferencias de los resultados. El análisis post-hoc indica que existen diferencias estadísticamente significativas entre el valor del Estimador05 y el resto de estimadores lo que confirma lo observado en las gráficas de los resultados, de manera similar a lo que ocurría con el porcentaje de aplicaciones ejecutadas en plazo.

El análisis ANOVA para el tiempo medio de espera (figura 5.30) indica que la tasa de llegada de aplicaciones (el tamaño del efecto es del 14 % - $\omega^2 = 0{,}149$) y el estimador del tiempo de procesamiento (el tamaño del efecto es del 66 % - $\omega^2 = 0{,}664$) tienen una influencia estadísticamente significativa en las diferencias de los resultados. El análisis post-hoc indica que existen diferencias estadísticamente significativas entre el valor del Estimador05 y el resto de estimadores, pero no muestra diferencias significativas entre las tasas de llegada de aplicaciones.

El análisis ANOVA para el coste medio por aplicación ejecutada en plazo (figura 5.31) indica que solo el estimador del tiempo de procesamiento (el tamaño del efecto es del 69 % - $\omega^2 = 0{,}692$) tienen una influencia estadísticamente significativa en las diferencias de los resultados. El análisis post-hoc indica que existen diferencias estadísticamente significativas entre el valor del Estimador05 y el resto de estimadores.

Los resultados indican que los estimadores Chebyshev, BosscheChebyshev,

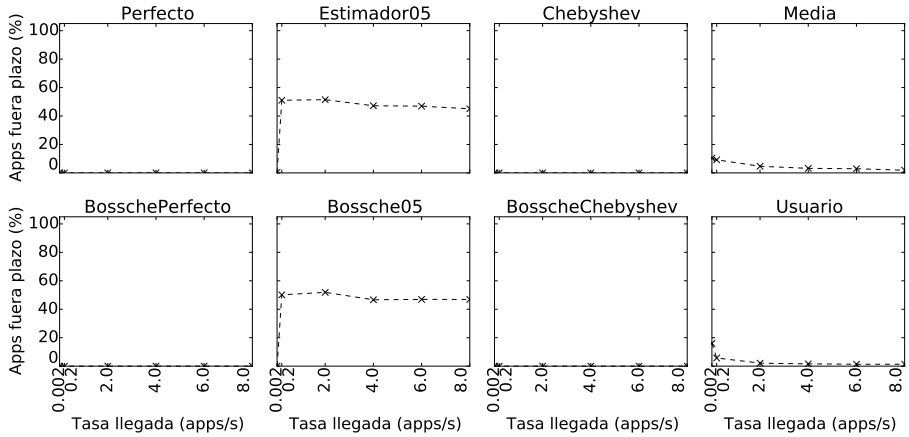



Figura 5.29: Carga Weibull, análisis por ratio de llegada, tiempo de aprovisionamiento 0 segundos, porcentaje de aplicaciones fuera de plazo

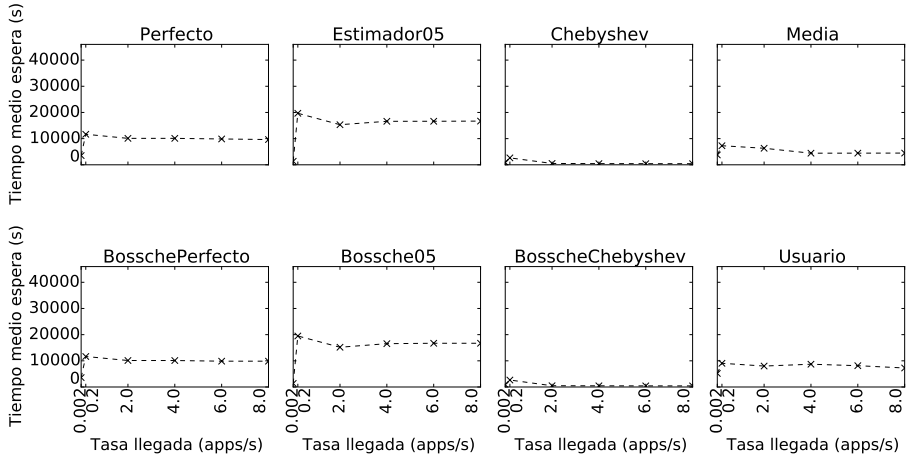



Figura 5.30: Carga Weibull, análisis por ratio de llegada, tiempo de aprovisionamiento 0 segundos, tiempo medio de espera por aplicación

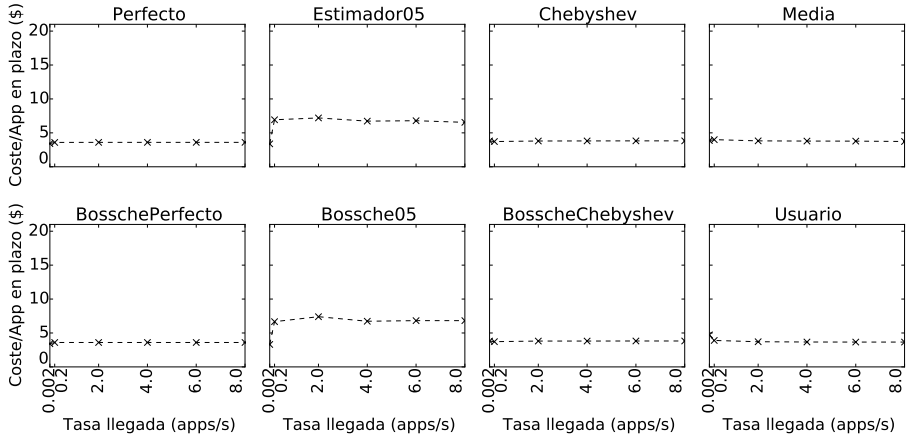



Figura 5.31: Carga Weibull, análisis por ratio de llegada, tiempo de aprovisionamiento 0 segundos, coste por aplicación ejecutada en plazo

Media y Usuario mejoran el número de aplicaciones ejecutadas en plazo a medida que el ratio de llegada de aplicaciones se incrementa. Además, se observa que a medida que el ratio de llegada de aplicaciones se incrementa, también se incrementa el número de instancias de máquinas virtuales solicitadas (ver figura 5.32). Esto significa que más trabajos son ejecutados en nuevas instancias de máquinas virtuales y que por tanto el error en las estimaciones de los tiempos de procesamiento tiene menor impacto.

El análisis ANOVA para el número de instancias de máquinas virtuales iniciadas (figura 5.32) indica que la tasa de llegada de aplicaciones (el tamaño del efecto es del 11 % - $\omega^2 = 0{,}113$) y el estimador del tiempo de procesamiento (el tamaño del efecto es del 78 % - $\omega^2 = 0{,}788$) tienen una influencia estadísticamente significativa en las diferencias de los resultados. El análisis post-hoc indica que existen diferencias estadísticamente significativas entre el valor del Estimador05 y el resto de estimadores, pero no muestra diferencias significativas entre las tasas de llegada de aplicaciones.

Al contrario de lo que ocurría en los escenarios de análisis por límite temporal con el factor de límite temporal, en este escenario de simulación el ratio de llegada de aplicaciones no se ha encontrado como un factor estadísticamente significativo sobre los resultados, dejando como único factor relevante el estimador de los tiempos de procesamiento.

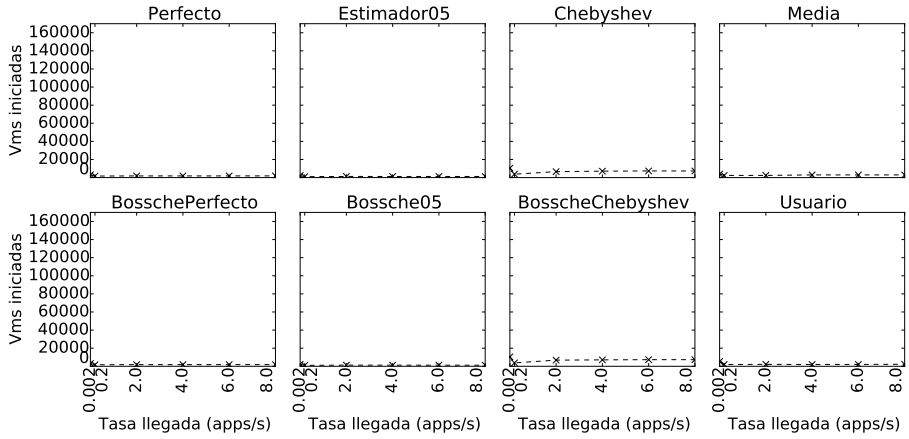



Figura 5.32: Carga Weibull, análisis por ratio de llegada, tiempo de aprovisionamiento 0 segundos, instancias de máquinas virtuales iniciadas

### 5.3.2. Carga Weibull con tiempo de aprovisionamiento de 100 segundos

Las figuras 5.33, 5.34, 5.35, 5.36 y 5.37 muestran los resultados de la simulación empleando la carga de trabajo basada en la distribución Weibull y con tiempos aprovisionamiento de instancias de máquinas virtuales de 100 segundos. La tabla B.7 del anexo B contiene los resultados numéricos.

Como se puede observar en la figura 5.33, los planificadores que emplean el estimador perfecto (Perfecto y BosschePerfecto) obtienen peores resultados que los planificadores Chebyshev, BosscheChebyshev, Media y Usuario cuando el ratio de llegada de aplicaciones es mayor de 2 aplicaciones por segundo. Los resultados de la simulación muestran que los planificadores que emplean el estimador Perfecto no ejecutan las aplicaciones en tiempo cuando los trabajos son los primeros trabajos a ejecutar en nuevas instancias de máquinas virtuales. El resto de estimadores no sufren tanto este problema dado que la sobreestimación compensa el tiempo de aprovisionamiento desconocido. El uso del estimador Estimador05 obtiene los peores resultados al igual que en el escenario de simulación anterior. El análisis ANOVA para el porcentaje de aplicaciones ejecutadas en plazo indica que solo el estimador del tiempo de procesamiento (el tamaño del efecto es del 42 % - $\omega^2 = 0{,}428$) tienen una influencia estadísticamente sig-

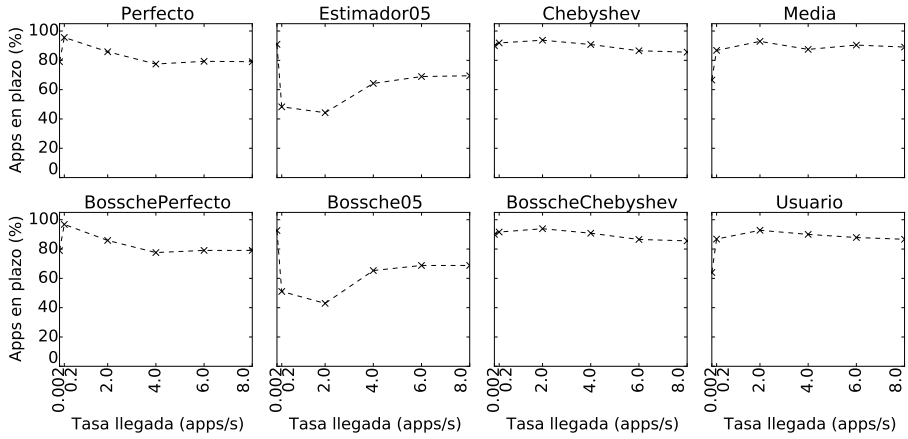



Figura 5.33: Carga Weibull, análisis por ratio de llegada, tiempo de aprovisionamiento 100 segundos, porcentaje de aplicaciones ejecutadas en plazo

nificativa en las diferencias de los resultados. El análisis post-hoc indica que existen diferencias estadísticamente significativas entre el valor del Estimador05 y el resto de estimadores.

El análisis ANOVA para el porcentaje de aplicaciones ejecutadas fuera de plazo (figura 5.34) no muestra ninguna variable independiente con una influencia estadísticamente significativa sobre los resultados.

El análisis ANOVA para el tiempo medio de espera (figura 5.35) indica que la tasa de llegada de aplicaciones (el tamaño del efecto es del 22 % - $\omega^2$ = 0,223) y el estimador del tiempo de procesamiento (el tamaño del efecto es del 52 % - $\omega^2$ = 0,528) tienen una influencia estadísticamente significativa en las diferencias de los resultados. El análisis post-hoc indica que existen diferencias estadísticamente significativas entre los valores de la tasa de llegada 2 y 200 y entre el estimador Estimador05 y el resto de estimadores y entre el estimador de Chebyshev y el resto de estimadores. Esto refuerza las observaciones de las gráficas donde se aprecia que los tiempos de espera más altos se corresponden a los planificadores que emplean el estimador Estimador05 y los más bajos los que emplean el estimador de Chebyshev.

En cuanto al coste medio por aplicación ejecutada en plazo, la figura 5.36 muestra que todas las combinaciones de planificador y estimador obtienen un coste muy similar, salvo cuando se emplea el estimador Estimador05. Cabe

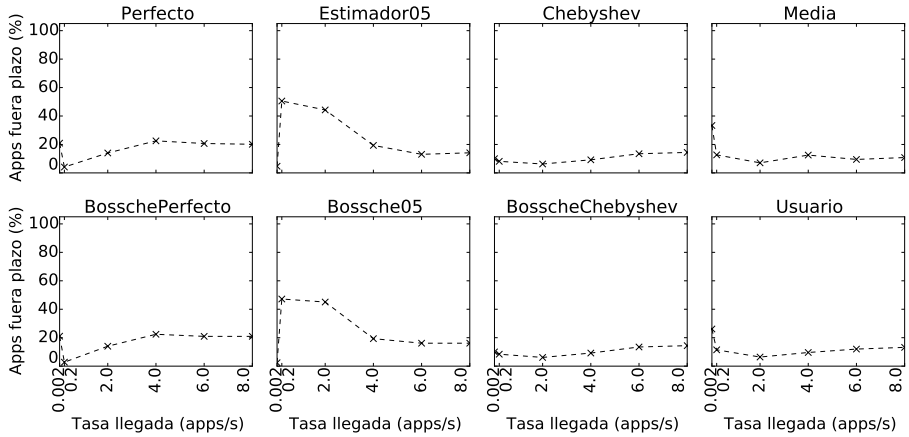



Figura 5.34: Carga Weibull, análisis por ratio de llegada, tiempo de aprovisionamiento 100 segundos, porcentaje de aplicaciones fuera de plazo

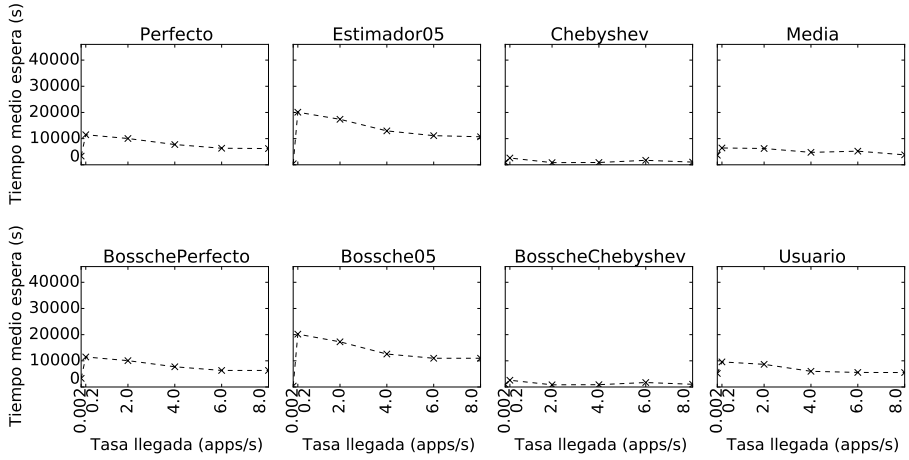



Figura 5.35: Carga Weibull, análisis por ratio de llegada, tiempo de aprovisionamiento 100 segundos, tiempo medio de espera por aplicación

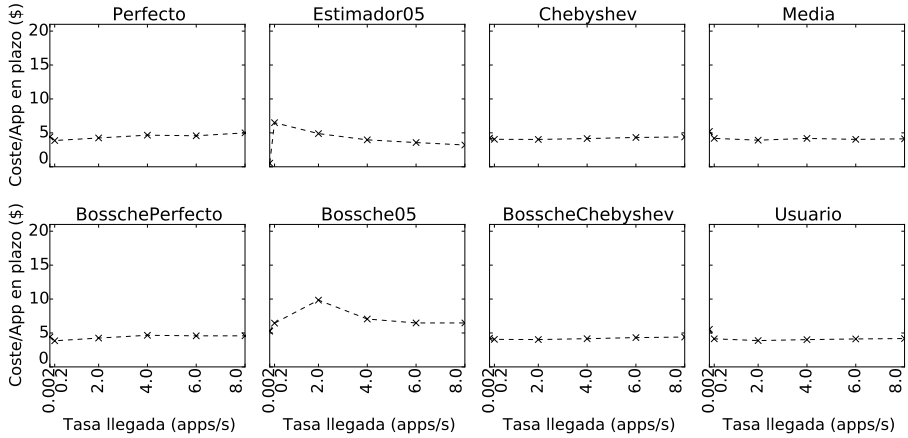



Figura 5.36: Carga Weibull, análisis por ratio de llegada, tiempo de aprovisionamiento 100 segundos, coste por aplicación ejecutada en plazo

destacar que con el estimador Estimador05 el planificador propuesto en este trabajo obtiene costes ligeramente inferiores a los que obtiene el planificador propuesto por Van den Bossche. Esta observación se ve apoyada por el análisis ANOVA que indica que el algoritmo de planificación es el único factor con un influencia estadísticamente significativa en las diferencias de los resultados, contando con un tamaño de efecto del 10 % ($\omega^2 = 0{,}105$).

El análisis ANOVA para el número de instancias de máquinas virtuales iniciadas (figura 5.37) indica que la tasa de llegada de aplicaciones (el tamaño del efecto es del 17 % - $\omega^2 = 0{,}177$) y el estimador del tiempo de procesamiento (el tamaño del efecto es del 61 % - $\omega^2 = 0{,}611$) tienen una influencia estadísticamente significativa en las diferencias de los resultados. El análisis post-hoc indica que existen diferencias estadísticamente significativas entre el valor del Estimador05 y el resto de estimadores y el estimador de Chebyshev y el resto de estimadores, pero no muestra diferencias significativas entre las tasas de llegada de aplicaciones.

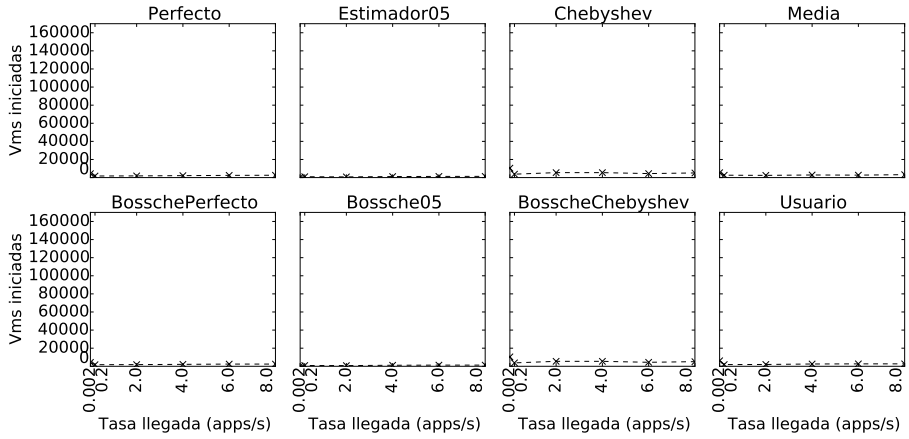



Figura 5.37: Carga Weibull, análisis por ratio de llegada, tiempo de aprovisionamiento 100 segundos, instancias de máquinas virtuales iniciadas

### 5.3.3. Carga Weibull para cualquier tiempo de aprovisionamiento

En esta sección se analizan los resultados de todas las simulaciones que emplean la carga basada en una distribución Weibull para cualquier tiempo de aprovisionamiento de nuevas instancias de máquinas virtuales. Para realizar este análisis se han unido los resultados numéricos presentados en las tablas B.6 y B.7 del anexo B.

La tabla 5.7 muestra los tamaños de los efectos de las variables independientes sobre las diferentes variables dependientes para aquellos casos en los que la variable independiente es estadísticamente significativa y cuando el análisis post-hoc posterior es capaz de encontrar niveles con diferencias.

Los resultados del análisis indican que la tasa de llegada de aplicaciones solo es un factor determinante desde el punto de vista estadístico para el tiempo medio de espera y para el número de instancias de máquinas virtuales iniciadas, aunque con un tamaño de efecto muy discreto y siempre inferior del 20 %. Sin embargo el estimador del tiempo de procesamiento sí es un factor muy relevante, con tamaños de efecto siempre superiores al 30 % y llegando en el caso de algunas variables a valores cercanos al 70 %. Estos resultados contrastan con los obtenidos para el mismo tipo de carga en el análisis por límite temporal, donde

| | Tasa llegadas | Aprovisionamiento | Estimador | Algoritmo |
|---|---|---|---|---|
| % en plazo | | 0.046 | 0.555 | |
| % fuera plazo | | | 0.430 | |
| Tiempo de espera | 0.180 | | 0.602 | |
| Coste | | | 0.321 | 0.024 |
| Núm. de instancias | 0.146 | | 0.693 | |

Tabla 5.7: Análisis ANOVA para la carga Weibull para cualquier tiempo de aprovisionamiento - Tamaño del efecto

el efecto del estimador era muy limitado. Al igual que en el análisis por límite temporal no se aprecian diferencias estadísticamente significativas entre los dos algoritmos de planificación comparados.

### 5.3.4. Carga Normal con tiempo de aprovisionamiento de 100 segundos

Las figuras 5.38 , 5.39 , 5.40 , 5.41 y 5.42 muestran los resultados de la simulación empleando una carga de trabajo basada en una distribución Normal y con el tiempo de aprovisionamiento de nuevas instancias de máquinas virtuales fijado a 100 segundos. La tabla B.8 del anexo B contiene los resultados numéricos.

En este escenario las combinaciones de planificador y estimador Perfecto, BosschePerfecto, Chebyshev, BosscheChebyshev y Usuario son capaces de alcanzar un 100 % o casi un 100 % de aplicaciones ejecutadas en plazo independientemente de la tasa de llegada de aplicaciones (figura 5.38). El análisis ANOVA para el porcentaje de aplicaciones ejecutadas en plazo confirma las percepciones mostradas en las gráficas e indica que el estimador (el tamaño del efecto es del 78 % - $\omega^2 = 0{,}787$) es el único factor que tiene una influencia estadísticamente significativa en las diferencias de los resultados. El análisis post-hoc indica que existen diferencias estadísticamente significativas entre el valor del Estimador05 y el resto de estimadores.

El análisis ANOVA para el porcentaje de aplicaciones ejecutadas fuera de plazo (figura 5.39) muestra un resultado muy similar al del porcentaje de aplicaciones ejecutadas en plazo. El estimador es el único factor con una influencia estadísticamente significativa con un tamaño de efecto del 78 % ($\omega^2 = 0{,}789$). El análisis post-hoc indica que existen diferencias estadísticamente significativas

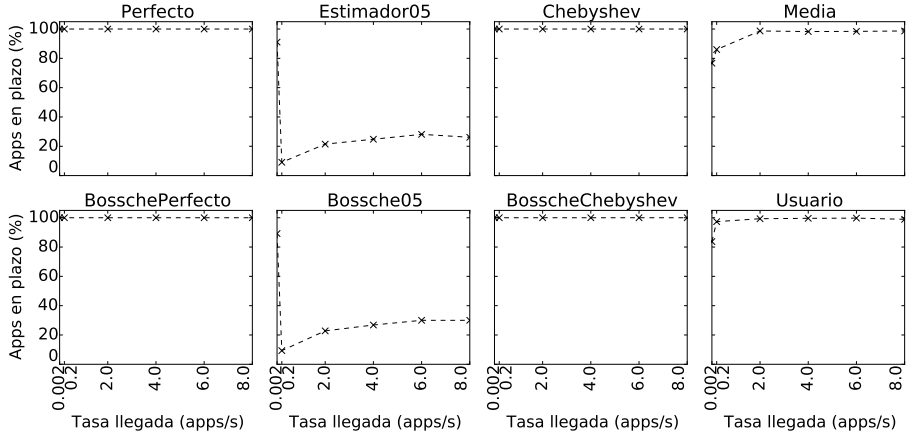



Figura 5.38: Carga normal, análisis por ratio de llegada, tiempo de aprovisionamiento 100 segundos, porcentaje de aplicaciones ejecutadas en plazo

entre el valor del Estimador05 y el resto de estimadores.

El análisis ANOVA para el tiempo medio de espera (figura 5.40) indica que la tasa de llegada de aplicaciones (el tamaño del efecto es del 4 % - $\omega^2 = 0{,}046$) y el estimador del tiempo de procesamiento (el tamaño del efecto es del 80 % - $\omega^2 = 0{,}807$) tienen una influencia estadísticamente significativa en las diferencias de los resultados. El análisis post-hoc indica que existen diferencias estadísticamente significativas entre el valor del Estimador05 y el resto de estimadores y el estimador de Chebyshev y el resto de estimadores, pero no muestra diferencias significativas entre las tasas de llegada de aplicaciones.

El análisis ANOVA para el coste medio por aplicación ejecutada en plazo (figura 5.41) indica que la tasa de llegada de aplicaciones (el tamaño del efecto es del 9 % - $\omega^2 = 0{,}094$) y el estimador del tiempo de procesamiento (el tamaño del efecto es del 42 % - $\omega^2 = 0{,}421$) tienen una influencia estadísticamente significativa en las diferencias de los resultados. El análisis post-hoc indica que existen diferencias estadísticamente significativas entre el valor del Estimador05 y el resto de estimadores, pero no muestra diferencias significativas entre las tasas de llegada de aplicaciones.

El análisis ANOVA para el número de instancias de máquinas virtuales iniciadas (figura 5.42) indica que la tasa de llegada de aplicaciones (el tamaño del efecto es del 9 % - $\omega^2 = 0{,}097$) y el estimador del tiempo de procesamiento

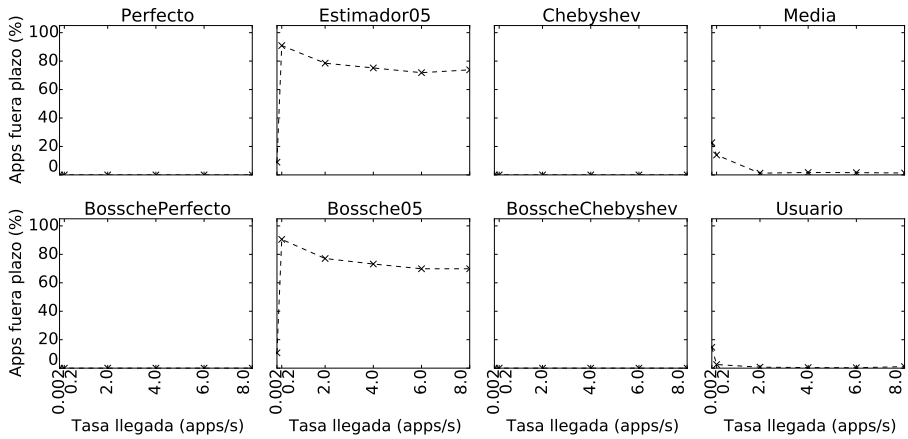



Figura 5.39: Carga normal, análisis por ratio de llegada, tiempo de aprovisionamiento 100 segundos, porcentaje de aplicaciones fuera de plazo

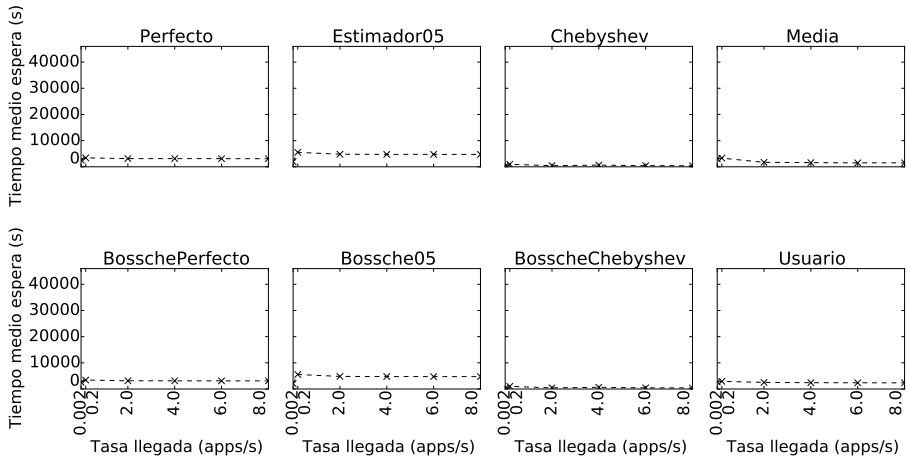



Figura 5.40: Carga normal, análisis por ratio de llegada, tiempo de aprovisionamiento 100 segundos, tiempo medio de espera por aplicación

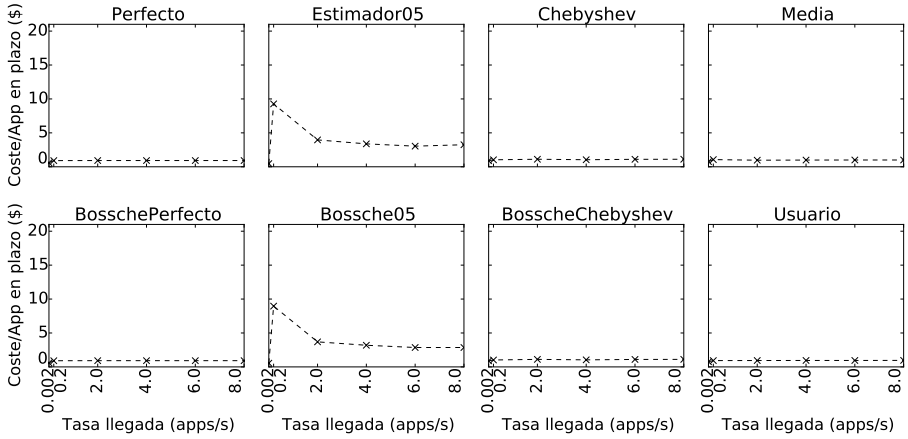



Figura 5.41: Carga normal, análisis por ratio de llegada, tiempo de aprovisionamiento 100 segundos, coste por aplicación ejecutada en plazo

(el tamaño del efecto es del 84 % - $\omega^2 = 0{,}844$) tienen una influencia estadísticamente significativa en las diferencias de los resultados. El análisis post-hoc indica que existen diferencias estadísticamente significativas entre el valor del Estimador05 y el resto de estimadores y el estimador de Chebyshev y el resto de estimadores, pero no muestra diferencias significativas entre las tasas de llegada de aplicaciones.

### 5.3.5. Análisis independiente de la carga y el tiempo de aprovisionamiento

En esta sección se presenta el análisis ANOVA realizado sobre los resultados de todos los escenarios de simulación relativos al análisis por ratio de llegada de trabajos, independientemente de la carga de trabajo y del tiempo de aprovisionamiento de nuevas instancias de máquinas virtuales. Estos dos factores se convierten en variables independientes del modelo ANOVA empleando la ecuación 5.5. Para realizar este análisis se han unido los resultados numéricos presentados en las tablas B.6, B.7 y B.8 del anexo B.

La tabla 5.8 muestra los tamaños de los efectos de las variables independientes sobre las diferentes variables dependientes para aquellos casos en los que la variable independiente es estadísticamente significativa y cuando el análisis

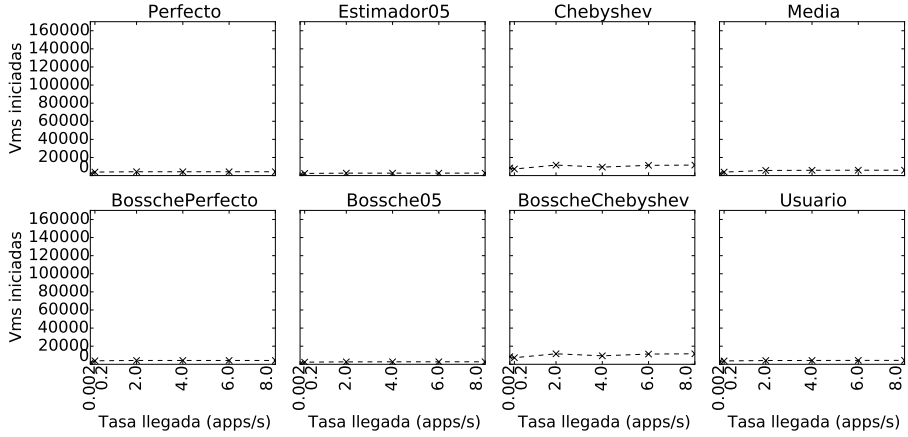



Figura 5.42: Carga normal, análisis por ratio de llegada, tiempo de aprovisionamiento 100 segundos, instancias de máquinas virtuales iniciadas

post-hoc posterior es capaz de encontrar niveles con diferencias.

Como se puede observar solo el tipo de carga de trabajo, la tasa de llegada de aplicaciones y el estimador de los tiempos de procesamiento tienen un efecto estadísticamente significativo sobre las diferentes variables dependientes que definen los resultados de los escenarios de simulación. Sin embargo, los tamaños de efecto indican que solo el tipo de carga en el caso del coste medio por aplicación ejecutada en plazo y el estimador en todos los casos tienen un efecto que se podría considerar apreciable. Estos resultados contrastan con los obtenidos para el análisis por límite temporal y ponen de manifiesto que el efecto de los

| | Carga | Tasa llegadas | Aprov. | Estimador | Algoritmo |
|---|---|---|---|---|---|
| % en plazo | | | | 0.609 | |
| % fuera plazo | | | | 0.537 | |
| Tiempo de espera | 0.112 | 0.115 | | 0.458 | |
| Coste | 0.395 | | | 0.220 | |
| Núm. de instancias | 0.084 | | | 0.672 | |

Tabla 5.8: Análisis ANOVA para el análisis por ratio de llegada independientemente de la carga y del tiempo de aprovisionamiento - Tamaño del efecto

estimadores de los tiempos de procesamiento varía en función del escenario de uso del planificador.

## 5.4. Discusión

Las simulaciones llevadas a cabo muestran que, en general, es posible obtener planificaciones adecuadas en términos de calidad de servicio (aplicaciones ejecutadas en plazo) y de coste económico empleando el planificador propuesto en este trabajo.

La tabla 5.9 resume los factores (variables independientes en los escenarios de simulación) que afectan de manera estadísticamente significativa a cada una de las variables dependientes que definen los resultados de los 8 escenarios simulados mas los 5 escenarios compuestos con fines de análisis. Se puede observar que los factores más determinantes en los resultados del análisis por límite temporal son el factor del límite temporal y el estimador. En el análisis por ratio de llegada el factor más determinante es el estimador. El algoritmo de planificación solo tiene un impacto estadísticamente significativo sobre el coste en dos escenarios de simulación.

Los resultados muestran que el algoritmo propuesto obtiene resultados iguales o mejores que el algoritmo propuesto por Van den Bossche [15] en todos los escenarios que emplean la carga de trabajo basada en una distribución Weibull, aunque como ya se ha visto las diferencias no son estadísticamente significativas. El planificador de este trabajo consigue una mejora del 12,4 % en términos de aplicaciones ejecutadas en plazo para el escenario de análisis por límite temporal, con un tiempo de aprovisionamiento de nuevas instancias de máquinas virtuales de 100 segundos, un factor de límite temporal de 2 y cuando se comparan ambos algoritmos empleando el estimador perfecto.

Cuando se emplea la carga de trabajo basada en una distribución Normal y un tiempo de aprovisionamiento de 100 segundos el algoritmo de Van den Bossche obtiene mejores resultados cuando se emplea el estimador Estimador05 y el estimador Perfecto. En todos los casos, las mejoras del algoritmo de Van den Bossche están por debajo del 5,6 % en términos de aplicaciones ejecutadas en plazo.

Estos resultados muestran que el tipo de carga de trabajo tiene un impacto significativo en el rendimiento de los algoritmos de planificación. Los resultados de la tabla 5.9 muestran, por ejemplo, que el tiempo de aprovisionamiento de nuevas instancias de máquinas virtuales tiene un impacto estadísticamente significativo en el análisis por límite temporal cuando se emplea la carga basada

| Análisis | Carga Aprov. | En plazo | Fuera plazo | Espera | Coste | Instancias |
|---|---|---|---|---|---|---|
| Límite temporal | Weibull - 0s | Factor Est. | Factor Est. | Factor Est. | Factor | Factor Est. |
| | Weibull - 100 s | Factor | Factor Est. | Factor Est. | Factor Alg. | Factor Est. |
| | Weibull | Factor Aprov. Est. | Factor Aprov. Est. | Factor Est. | Factor Aprov. Alg. | Factor Est. |
| | Normal - 0s | Factor | Factor Est. | Factor Est. | Factor | Factor Est. |
| | Normal - 100s | Factor | Factor Est. | Factor Est. | Factor | Factor Est. |
| | Normal | Factor Est. | Factor Est. | Factor Est. | Factor | Factor |
| | Google - 0s | Factor | | Factor | Factor | Factor |
| | Todos | Factor Aprov. Est. | Factor Aprov. Est. | Factor Est. | Factor | Factor Est. |
| Ratio llegada | Weibull - 0s | Est. | Est. | Est. | Est. | Est. |
| | Weibull - 100 s | Est. | | Ratio Est. | Alg. | Est. |
| | Weibull | Aprov. Est. | Est. | Ratio Est. | Est. Alg. | Ratio Est. |
| | Normal - 100s | Est. | Est. | Est. | Est. | Est. |
| | Todos | Est. | Est. | Carga Ratio Est. | Carga Est. | Carga Est. |

Tabla 5.9: Resumen del análisis ANOVA para los escenarios simulados y las uniones de resultados de los escenarios: factores estadísticamente significativos

en un distribución Weibull, pero no cuando se emplea la carga basada en una Normal.

La principal ventaja del estimador basado en la desigualdad de Chebyshev es que puede ser empleado con cargas de trabajo cuyos tiempos de procesamiento sigan cualquier distribución de probabilidad. Otra ventaja es que el grado de pesimismo de las estimaciones puede ser regulado mediante el ajuste del parámetro $k$ de la desigualdad. Dado que menos pesimismo puede implicar un mayor número de errores en las estimaciones pero menos coste en infraestructura, este pesimismo puede ser empleado para regular el equilibrio entre la probabilidad de que una aplicación sea marcada como inviable y el sobre-aprovisionamiento de recursos en el Cloud público.

Comparando los resultados de la carga de trabajo basada en una distribución Normal con los resultados cuando la carga de trabajo se basa en una Weibull, cabe destacar que el coste por cada aplicación ejecutada en plazo no puede ser directamente relacionado con el número de instancias de máquinas virtuales empleadas. El coste total de ejecutar una carga de trabajo depende del número de instancias empleadas y del tiempo total que estas instancias son empleadas. Por ejemplo, con un ratio de llegada de 2 aplicaciones por segundo y un tiempo de aprovisionamiento de 100 segundos el planificador Perfecto necesita necesita 13845 horas de computación de 4230 instancias con la carga Normal (coste total de 692,25$) y 54678 horas de computación de 1850 instancias con la carga Weibull (coste total de 2733,9$).

El estimador basado en la desigualdad de Chebyshev siempre obtiene mejores resultados en cuanto a aplicaciones ejecutadas en plazo que el estimador de usuario en todas las simulaciones. El estimador de usuario solo obtiene mejores resultados que el estimador de media en la carga basada en la distribución Normal. Las diferencias en los resultados entre el estimador de Usuario y el estimador de media cuando se emplea la carga basada en la distribución Weibull y la carga basada en la distribución Normal demuestra que estos estimadores se ven fuertemente afectados por el tipo de carga de trabajo. La aplicación del estimador de Chebyshev al planificador de Van den Bossche también obtiene buenos resultados. El algoritmo obtiene mejores porcentajes de aplicaciones ejecutadas en plazo que cuando se emplea el mecanismo de estimación de tiempos de procesamiento propuesto por el mismo autor para todos los escenarios de simulación, excepto para la carga basada en una distribución Normal y un factor de límite temporal de 2.

Los experimentos de simulación han sido llevados a cabo sin tener en cuenta el tiempo y el coste de transferir los datos necesarios para ejecutar las cargas de trabajo entre el Cloud privado y el Cloud público. Esta simplificación no

condiciona la validez de los resultados finales dado que el tiempo de uso de red puede ser fácilmente medido e incluido tanto en el planificador como en los estimadores de los tiempos de procesamiento. En este escenario los estimadores producirían estimaciones de los tiempos globales de servicio de las tareas y no solo de los tiempos de procesamiento en CPU. Además, debe tenerse en cuenta que el planificador presentado en este trabajo esta diseñado para aquellos casos de uso en los que los datos a analizar dentro de la carga de trabajo están disponibles desde cualquier instancia del Cloud mediante una conexión a una base de datos remota o mediante un sistema de ficheros distribuido en red. En este tipo de soluciones el acceso a lo datos siempre es realizado mediante conexiones de red y los nodos de computación no tienen datos locales. En este escenario todos los nodos de procesamiento (las instancias de las máquinas virtuales) acceden a los datos a través de conexiones de red y la penalización en tiempo y coste de la transmisión de los datos es similar en todos los escenarios. Este tipo de sistemas incluyen arquitecturas escalables y de alta disponibilidad que emplean algoritmos de procesamiento sin estado y fuentes de datos en forma de bases de datos relacionales replicadas o almacenes de datos NoSQL. Debe tenerse en cuenta que aunque los costes de ancho de banda y por unidad de datos son ligeramente diferentes entre los proveedores de Cloud públicos estos costes son pequeños en comparación con los costes de los tiempos de procesamiento.

El precio de la transferencia de datos por red en los proveedores de Cloud públicos depende del origen y destino de los datos. Mientras que la mayoría de los proveedores ofrecen descarga de datos gratuita desde Internet hacia sus infraestructuras, la transferencia de datos salientes se cobra en función del destino de los datos. Transferir datos entre nodos en la misma zona o región geográfica suele ser gratuito o muy barato mientras que transferir datos entre diferentes regiones tiene un coste que depende del origen y destino. Modelar este esquema de precios de transferencia de datos sería complejo porque existen múltiples posibilidades dependiendo de los nodos de procesamiento (Cloud privado, mismo Cloud público que los nodos de datos, diferente Cloud público del que alberga los nodos de datos, etc.) y los nodos que albergan los datos (instancias en el Cloud privado o en en el Cloud público). Aunque nuestro modelo de Cloud puede ser extendido para incorporar estos costes, un modelo para simular de manera precisa los costes de las transferencias de red sería extremadamente complejo y se escapa del ámbito de este trabajo. Desde el punto de vista del sistema aquí propuesto, si una aplicación debe acceder a datos de almacenes remotos el tiempo de descarga de datos se considera incluido en el tiempo de procesamiento. Se asume por tanto que el tiempo de acceso a los datos se incluye en el tiempo de procesamiento de los trabajos y que por tanto está también incluido en las

estimaciones del tiempo de procesamiento.

El tiempo de aprovisionamiento de una nueva instancia de máquina virtual en el Cloud público es simulado pero no es tenido en cuenta por los algoritmos de planificación. Al igual que en el caso del tiempo de transferencia de red, este tiempo puede ser fácilmente introducido en el algoritmo añadiendo un estimador del tiempo de aprovisionamiento basado en las mismas estrategias que se han empleado en los estimadores del tiempo de procesamiento. El efecto de este tiempo de aprovisionamiento es importante tal y como se ha visto en los resultados de las simulaciones, pero está siendo mitigado por los proveedores de Cloud públicos mediante la introducción de tecnologías hardware y software que permiten disminuir los tiempos de aprovisionamiento de instancias de máquinas virtuales. Este es un aspecto muy importante de las soluciones Cloud dado que las ventajas de la elasticidad pueden verse muy reducidas si dicha elasticidad no puede responder de manera rápida a los incrementos o decrementos de la carga de trabajo.

La principal limitación de la estrategia de planificación y los estimadores del tiempo de procesamiento propuestos está relacionada con los escenarios en los que los plazos límite de ejecución de las aplicaciones son muy ajustados en comparación con el tiempo de procesamiento necesario. El estimador que obtiene los mejores resultados en la mayoría de los escenarios, el estimador basado en la desigualdad de Chebyshev, tiende a marcar demasiados trabajos como inviables cuando el plazo de ejecución es muy ajustado. Esta situación puede ser solventada ajustando dinámicamente la estimación calculada mediante el parámetro $k$ de la desigualdad de Chebyshev en función de lo ajustado del plazo de ejecución.

En este trabajo se ha demostrado que una estrategia de planificación que no emplea estimaciones de los tiempos de procesamiento como entrada por parte de los usuarios es viable para la planificación de bolsas de tareas con plazo de ejecución. Los resultados indican que las estimaciones de los tiempos de procesamiento basados en las medidas de los tiempos de procesamiento de trabajos previamente ejecutados, en combinación con la estrategia de planificación propuesta, obtienen buenos resultados en términos de aplicaciones ejecutadas en plazo. Los resultados varían dependiendo de las estructura de la carga de trabajo, lo ajustado del plazo de ejecución o la tasa de llegada de aplicaciones al sistema, pero el estimador de los tiempos de procesamiento basado en la desigualdad de Chebyshev es capaz de obtener resultados tan buenos como el estimador Perfecto en muchas circunstancias o, en algunos escenarios concretos, incluso mejores. Una clara ventaja de este estimador es que no requiere de datos proporcionados por los usuarios antes de la ejecución de las aplicaciones.

# 6

# Conclusiones

El uso de infraestructuras de Cloud híbridas para la ejecución de cargas de trabajo formadas por bolsas de tareas con límite temporal de ejecución introduce el problema de gestionar las instancias de máquinas virtuales y la planificación de los trabajos de una manera eficiente en costes, al mismo tiempo que se respetan los plazos temporales. Esta tesis presenta una estrategia de planificación capaz de resolver este problema de optimización de manera aproximada mediante un algoritmo de planificación complementado con el uso de estimadores de los tiempos de procesamiento basados en datos muestrales. El uso de estos estimadores elimina la necesidad de caracterizar la carga de trabajo antes de su ejecución y permite la implementación de planificadores autónomos y adaptables, principal limitación de los trabajos existentes en el estado del arte.

El algoritmo de planificación está compuesto por dos subplanificadores, uno para el Cloud privado y otro para el Cloud público, que tratan de maximizar el número de aplicaciones ejecutadas en plazo (objetivo primario) y minimizar el coste de la infraestructura pública (objetivo secundario). En ambos subplanificadores las aplicaciones son ordenadas según una política EDF. Los estimadores

de los tiempos de procesamiento son empleados para construir una planificación tentativa, de manera que si el subplanificador del Cloud privado no es capaz de cumplir un límite temporal de ejecución la aplicación es enviada al Cloud público. Si el planificador del Cloud público no es capaz de cumplir el límite temporal de una aplicación con la infraestructura contratada, puede aprovisionar nuevas instancias de máquinas virtuales tratando de minimizar el coste o descartar completamente la aplicación.

Se han propuesto cuatro estimadores de los tiempos de procesamiento: un estimador perfecto con fines de validación, un estimador que trata de imitar las estimaciones proporcionadas por los usuarios y los estimadores de media y de Chebyshev. Estos dos últimos estimadores emplean los datos de las tareas previamente ejecutadas y emplean una media móvil y el teorema de Chebyshev respectivamente para producir sus estimaciones.

El proceso de validación de la estrategia de planificación se ha realizado comparando el algoritmo de planificación aquí propuesto con otro existente en el estado del arte y empleando los cuatro estimadores propuestos más uno del estado del arte. Los elementos anteriores se han combinado en 8 estrategias de planificación que han sido comparadas con tres cargas de trabajo diferentes en 8 escenarios que varian la carga de trabajo, el tiempo de aprovisionamiento de las instancias de máquinas virtuales, el límite temporal y la tasa de llegada de las aplicaciones. Los resultados obtenidos en cada escenario se han cuantificado con cinco métricas: aplicaciones ejecutadas en plazo, aplicaciones ejecutadas fuera de plazo, tiempo medio de espera, coste total y número de instancias de máquinas virtuales iniciadas.

Se ha demostrado que la estrategia de planificación propuesta y el uso de estimadores de los tiempos de procesamiento basados en medidas es capaz de funcionar correctamente en múltiples situaciones. Los resultados de las simulaciones muestran que el estimador empleado afecta de manera estadísticamente significativa en 7 de los 8 escenarios y a casi todas las métricas, mientras que el algoritmo tiene un impacto estadísticamente significativo solo en 2 escenarios y solo sobre la métrica de coste (objetivo secundario de la planificación). En términos del número de aplicaciones ejecutadas en plazo (objetivo primario de la planificación) no se han encontrado diferencias estadísticamente significativas entre el algoritmo aquí propuesto y el algoritmo del estado del arte empleado como referencia. Estos resultados demuestran el impacto de los estimadores de los tiempos de procesamiento sobre el funcionamiento de los algoritmos de planificación y ponen de manifiesto que son un aspecto más importante de lo que hasta ahora se había tenido en cuenta en el estado del arte.

El estimador basado en la desigualdad de Chebyshev obtiene los mejores

resultados en términos del número de aplicaciones ejecutadas en plazo cuando el límite temporal de ejecución de las aplicaciones es mayor de 2 veces el tiempo de procesamiento base de los trabajos. La principal ventaja de este estimador es que no necesita información previa o realizar asunciones acerca de la naturaleza de la carga de trabajo, lo que supone una aportación sobre el estado del arte. La estrategia de planificación compuesta por el estimador basado en la desigualdad de Chebyshev y el algoritmo propuesto en este trabajo es capaz de obtener resultados similares (y en la mayoría de los escenarios estadísticamente indistinguibles) a los del algoritmo de planificación empleado como referencia, pero sin contar de antemano con ninguna información de la carga de trabajo.

El trabajo futuro incluye el análisis del algoritmo de planificación en un modelo de Cloud más detallado que incluya aspectos relacionados con el coste y tiempo de la transferencia de datos. Otra futura vía de investigación es la adaptación dinámica de la parametrización del estimador de Chebyshev para aquellos escenarios con plazos de ejecución muy ajustados.

# A

# Publicaciones

Las siguientes publicaciones se han derivado del trabajo de investigación asociado a esta tesis:

- Autonomic Scheduling of Deadline-Constrained Bag of Tasks in Hybrid Clouds. Víctor Peláez, Antonio Campos, Daniel F. García, Joaquín Entrialgo. 2016 International Symposium on Performance Evaluation of Computer and Telecommunication Systems.

  https://doi.org/10.1109/SPECTS.2016.7570526

- Online Scheduling of Deadline-Constrained Bag of Task Workloads on Hybrid Clouds. Víctor Peláez, Antonio Campos, Daniel F. García, Joaquín Entrialgo. Concurrency and Computation: Practice and Experience.

  http://dx.doi.org/10.1002/cpe.4639

# B

# Tablas de resultados

Este apéndice contiene las tablas de resultados de las simulaciones empleadas durante la evaluación presentada en el capítulo 5.

| Análisis | Carga | Aprov. | Factor | Algortimo | Estimador | En plazo | Fuera plazo | Espera | Coste | VMs |
|---|---|---|---|---|---|---|---|---|---|---|
| limite | weibull | 0 | 2 | pelaez | perfecto | 100.00 | 0.00 | 380.82 | 3.74 | 9091 |
| limite | weibull | 0 | 2 | pelaez | media | 75.33 | 23.07 | 938.97 | 5.11 | 11769 |
| limite | weibull | 0 | 2 | pelaez | chebyshev | 64.13 | 0.27 | 9.15 | 8.35 | 790 |
| limite | weibull | 0 | 2 | pelaez | usuario | 50.67 | 31.33 | 1470.35 | 7.59 | 12114 |
| limite | weibull | 0 | 3 | pelaez | perfecto | 100.00 | 0.00 | 1861.52 | 3.54 | 5381 |
| limite | weibull | 0 | 3 | pelaez | media | 88.53 | 11.47 | 2193.05 | 3.97 | 6078 |
| limite | weibull | 0 | 3 | pelaez | chebyshev | 100.00 | 0.00 | 138.22 | 4.02 | 13896 |
| limite | weibull | 0 | 3 | pelaez | usuario | 74.27 | 17.20 | 2928.69 | 4.90 | 7588 |
| limite | weibull | 0 | 4 | pelaez | perfecto | 100.00 | 0.00 | 3528.01 | 3.42 | 3142 |
| limite | weibull | 0 | 4 | pelaez | media | 89.20 | 10.53 | 3762.63 | 3.80 | 3849 |
| limite | weibull | 0 | 4 | pelaez | chebyshev | 100.00 | 0.00 | 880.09 | 3.81 | 10508 |
| limite | weibull | 0 | 4 | pelaez | usuario | 74.13 | 16.00 | 5186.99 | 4.72 | 5314 |
| limite | weibull | 0 | 5 | pelaez | perfecto | 100.00 | 0.00 | 6221.97 | 3.35 | 2055 |
| limite | weibull | 0 | 5 | pelaez | media | 87.73 | 12.13 | 6416.51 | 3.80 | 2936 |
| limite | weibull | 0 | 5 | pelaez | chebyshev | 100.00 | 0.00 | 2276.04 | 3.53 | 5396 |
| limite | weibull | 0 | 5 | pelaez | usuario | 82.27 | 10.40 | 7060.48 | 4.11 | 3351 |
| limite | weibull | 0 | 6 | pelaez | perfecto | 100.00 | 0.00 | 9816.48 | 3.28 | 1445 |
| limite | weibull | 0 | 6 | pelaez | media | 91.20 | 8.80 | 8215.42 | 3.67 | 1930 |
| limite | weibull | 0 | 6 | pelaez | chebyshev | 100.00 | 0.00 | 2900.44 | 3.47 | 4225 |
| limite | weibull | 0 | 6 | pelaez | usuario | 86.67 | 6.40 | 9510.85 | 3.84 | 2731 |
| limite | weibull | 0 | 7 | pelaez | perfecto | 100.00 | 0.00 | 16565.52 | 3.24 | 945 |
| limite | weibull | 0 | 7 | pelaez | media | 93.07 | 6.67 | 12009.15 | 3.51 | 1589 |
| limite | weibull | 0 | 7 | pelaez | chebyshev | 100.00 | 0.00 | 4132.17 | 3.38 | 2857 |
| limite | weibull | 0 | 7 | pelaez | usuario | 88.27 | 6.53 | 12428.86 | 3.71 | 2112 |
| limite | weibull | 0 | 8 | pelaez | perfecto | 100.00 | 0.00 | 29311.48 | 3.21 | 690 |
| limite | weibull | 0 | 8 | pelaez | media | 95.60 | 4.00 | 14852.19 | 3.38 | 1383 |
| limite | weibull | 0 | 8 | pelaez | chebyshev | 100.00 | 0.00 | 5507.80 | 3.35 | 2380 |
| limite | weibull | 0 | 8 | pelaez | usuario | 91.07 | 4.53 | 16006.47 | 3.54 | 1481 |
| limite | weibull | 0 | 9 | pelaez | perfecto | 100.00 | 0.00 | 43490.00 | 3.15 | 515 |
| limite | weibull | 0 | 9 | pelaez | media | 94.53 | 4.67 | 24842.64 | 3.35 | 745 |
| limite | weibull | 0 | 9 | pelaez | chebyshev | 100.00 | 0.00 | 7279.15 | 3.30 | 1876 |
| limite | weibull | 0 | 9 | pelaez | usuario | 92.53 | 3.60 | 17418.28 | 3.44 | 1093 |

(*Continúa en la página siguiente*)

| Análisis | Carga | Aprov. | Factor | Algortimo | Estimador | En plazo | Fuera plazo | Espera | Coste | VMs |
|---|---|---|---|---|---|---|---|---|---|---|
| limite | weibull | 0 | 2 | bossche | estimador05 | 33.47 | 66.53 | 100.97 | 5.00 | 1789 |
| limite | weibull | 0 | 3 | bossche | estimador05 | 98.00 | 2.00 | 616.72 | 2.92 | 3141 |
| limite | weibull | 0 | 4 | bossche | estimador05 | 99.87 | 0.13 | 1304.39 | 3.36 | 2175 |
| limite | weibull | 0 | 5 | bossche | estimador05 | 99.87 | 0.13 | 2696.20 | 3.31 | 1650 |
| limite | weibull | 0 | 6 | bossche | estimador05 | 100.00 | 0.00 | 4702.39 | 3.34 | 1371 |
| limite | weibull | 0 | 7 | bossche | estimador05 | 99.87 | 0.13 | 6673.59 | 3.27 | 1180 |
| limite | weibull | 0 | 8 | bossche | estimador05 | 99.73 | 0.27 | 10478.29 | 3.28 | 1002 |
| limite | weibull | 0 | 9 | bossche | estimador05 | 99.87 | 0.13 | 17160.93 | 3.21 | 863 |
| limite | weibull | 0 | 2 | bossche | perfecto | 100.00 | 0.00 | 380.51 | 3.74 | 9025 |
| limite | weibull | 0 | 3 | bossche | perfecto | 100.00 | 0.00 | 1811.91 | 3.55 | 5346 |
| limite | weibull | 0 | 4 | bossche | perfecto | 100.00 | 0.00 | 3568.99 | 3.42 | 3168 |
| limite | weibull | 0 | 5 | bossche | perfecto | 100.00 | 0.00 | 6376.00 | 3.32 | 2016 |
| limite | weibull | 0 | 6 | bossche | perfecto | 100.00 | 0.00 | 10404.66 | 3.27 | 1376 |
| limite | weibull | 0 | 7 | bossche | perfecto | 100.00 | 0.00 | 16377.81 | 3.24 | 994 |
| limite | weibull | 0 | 8 | bossche | perfecto | 100.00 | 0.00 | 29076.04 | 3.20 | 724 |
| limite | weibull | 0 | 9 | bossche | perfecto | 100.00 | 0.00 | 38197.14 | 3.16 | 601 |
| limite | weibull | 0 | 2 | bossche | chebyshev | 58.80 | 1.07 | 24.64 | 9.51 | 837 |
| limite | weibull | 0 | 3 | bossche | chebyshev | 100.00 | 0.00 | 138.46 | 4.02 | 13893 |
| limite | weibull | 0 | 4 | bossche | chebyshev | 100.00 | 0.00 | 881.62 | 3.81 | 10455 |
| limite | weibull | 0 | 5 | bossche | chebyshev | 100.00 | 0.00 | 2279.01 | 3.53 | 5377 |
| limite | weibull | 0 | 6 | bossche | chebyshev | 100.00 | 0.00 | 2901.55 | 3.47 | 4205 |
| limite | weibull | 0 | 7 | bossche | chebyshev | 100.00 | 0.00 | 4129.98 | 3.39 | 2853 |
| limite | weibull | 0 | 8 | bossche | chebyshev | 100.00 | 0.00 | 5565.05 | 3.35 | 2340 |
| limite | weibull | 0 | 9 | bossche | chebyshev | 100.00 | 0.00 | 7183.25 | 3.33 | 1851 |
| limite | weibull | 0 | 2 | pelaez | estimador05 | 34.00 | 66.00 | 96.63 | 5.00 | 1731 |
| limite | weibull | 0 | 3 | pelaez | estimador05 | 97.73 | 2.27 | 608.06 | 2.82 | 2994 |
| limite | weibull | 0 | 4 | pelaez | estimador05 | 100.00 | 0.00 | 1358.43 | 3.29 | 2199 |
| limite | weibull | 0 | 5 | pelaez | estimador05 | 100.00 | 0.00 | 2587.50 | 3.28 | 1651 |
| limite | weibull | 0 | 6 | pelaez | estimador05 | 100.00 | 0.00 | 4762.59 | 3.29 | 1370 |
| limite | weibull | 0 | 7 | pelaez | estimador05 | 99.60 | 0.27 | 6313.41 | 3.30 | 1208 |
| limite | weibull | 0 | 8 | pelaez | estimador05 | 100.00 | 0.00 | 9373.49 | 3.27 | 1049 |
| limite | weibull | 0 | 9 | pelaez | estimador05 | 100.00 | 0.00 | 17318.74 | 3.27 | 820 |

Tabla B.1: Resultados análisis por límite temporal, carga Weibull, tiempo de aprovisionamiento de 0 segundos

| Análisis | Carga | Aprov. | Factor | Algortimo | Estimador | En plazo | Fuera plazo | Espera | Coste | VMs |
|---|---|---|---|---|---|---|---|---|---|---|
| limite | weibull | 100 | 2 | pelaez | perfecto | 61.20 | 38.80 | 433.25 | 6.15 | 9326 |
| limite | weibull | 100 | 2 | pelaez | media | 45.07 | 52.13 | 1009.32 | 8.79 | 11238 |
| limite | weibull | 100 | 2 | pelaez | chebyshev | 48.53 | 3.60 | 14.61 | 15.75 | 872 |
| limite | weibull | 100 | 2 | pelaez | usuario | 33.33 | 48.67 | 1544.21 | 11.53 | 12047 |
| limite | weibull | 100 | 3 | pelaez | perfecto | 73.20 | 26.80 | 1868.89 | 4.91 | 6349 |
| limite | weibull | 100 | 3 | pelaez | media | 60.13 | 39.20 | 2383.74 | 6.48 | 5574 |
| limite | weibull | 100 | 3 | pelaez | chebyshev | 78.40 | 21.60 | 218.65 | 5.10 | 13551 |
| limite | weibull | 100 | 3 | pelaez | usuario | 56.13 | 35.33 | 2961.57 | 6.53 | 7992 |
| limite | weibull | 100 | 4 | pelaez | perfecto | 79.07 | 20.93 | 3327.76 | 4.41 | 4328 |
| limite | weibull | 100 | 4 | pelaez | media | 66.53 | 33.33 | 3668.20 | 5.21 | 5394 |
| limite | weibull | 100 | 4 | pelaez | chebyshev | 90.00 | 10.00 | 942.79 | 4.23 | 10431 |
| limite | weibull | 100 | 4 | pelaez | usuario | 64.00 | 26.13 | 5185.99 | 5.54 | 6090 |
| limite | weibull | 100 | 5 | pelaez | perfecto | 86.27 | 13.73 | 5578.09 | 3.94 | 3189 |
| limite | weibull | 100 | 5 | pelaez | media | 72.53 | 26.67 | 6253.82 | 4.69 | 4015 |
| limite | weibull | 100 | 5 | pelaez | chebyshev | 90.00 | 10.00 | 2266.99 | 4.03 | 7197 |
| limite | weibull | 100 | 5 | pelaez | usuario | 72.00 | 20.67 | 6942.18 | 4.79 | 4754 |
| limite | weibull | 100 | 6 | pelaez | perfecto | 89.47 | 10.53 | 8069.96 | 3.72 | 2473 |
| limite | weibull | 100 | 6 | pelaez | media | 80.00 | 18.53 | 7832.21 | 4.26 | 2864 |
| limite | weibull | 100 | 6 | pelaez | chebyshev | 90.00 | 10.00 | 2900.10 | 3.91 | 5153 |
| limite | weibull | 100 | 6 | pelaez | usuario | 78.80 | 14.00 | 9235.52 | 4.28 | 3518 |
| limite | weibull | 100 | 7 | pelaez | perfecto | 91.87 | 8.13 | 10864.77 | 3.58 | 1933 |
| limite | weibull | 100 | 7 | pelaez | media | 83.20 | 16.13 | 11040.99 | 4.10 | 2426 |
| limite | weibull | 100 | 7 | pelaez | chebyshev | 90.13 | 9.87 | 3915.93 | 3.82 | 3864 |
| limite | weibull | 100 | 7 | pelaez | usuario | 81.33 | 13.47 | 11992.71 | 4.07 | 2764 |
| limite | weibull | 100 | 8 | pelaez | perfecto | 93.33 | 6.67 | 13278.22 | 3.47 | 1520 |
| limite | weibull | 100 | 8 | pelaez | media | 86.53 | 12.53 | 12763.98 | 3.82 | 2086 |
| limite | weibull | 100 | 8 | pelaez | chebyshev | 92.53 | 7.47 | 5057.77 | 3.66 | 3089 |
| limite | weibull | 100 | 8 | pelaez | usuario | 85.07 | 10.53 | 14865.29 | 3.82 | 2022 |
| limite | weibull | 100 | 9 | pelaez | perfecto | 94.53 | 5.47 | 18904.97 | 3.36 | 1104 |
| limite | weibull | 100 | 9 | pelaez | media | 89.60 | 9.60 | 16153.71 | 3.96 | 1223 |
| limite | weibull | 100 | 9 | pelaez | chebyshev | 93.60 | 6.40 | 6262.21 | 3.56 | 2596 |
| limite | weibull | 100 | 9 | pelaez | usuario | 86.93 | 9.20 | 16603.78 | 3.69 | 1655 |

(*Continúa en la página siguiente*)

| Análisis | Carga | Aprov. | Factor | Algortimo | Estimador | En plazo | Fuera plazo | Espera | Coste | VMs |
|---|---|---|---|---|---|---|---|---|---|---|
| limite | weibull | 100 | 2 | bossche | estimador05 | 41.60 | 55.47 | 37.95 | 8.74 | 372 |
| limite | weibull | 100 | 3 | bossche | estimador05 | 89.60 | 4.27 | 191.89 | 5.39 | 390 |
| limite | weibull | 100 | 4 | bossche | estimador05 | 93.60 | 2.53 | 196.24 | 5.99 | 326 |
| limite | weibull | 100 | 5 | bossche | estimador05 | 94.93 | 1.47 | 795.88 | 5.18 | 315 |
| limite | weibull | 100 | 6 | bossche | estimador05 | 95.07 | 2.13 | 1925.64 | 4.17 | 243 |
| limite | weibull | 100 | 7 | bossche | estimador05 | 94.13 | 2.27 | 3677.40 | 4.22 | 282 |
| limite | weibull | 100 | 8 | bossche | estimador05 | 94.67 | 1.07 | 5589.33 | 5.80 | 306 |
| limite | weibull | 100 | 9 | bossche | estimador05 | 96.27 | 0.27 | 7403.60 | 6.17 | 244 |
| limite | weibull | 100 | 2 | bossche | perfecto | 48.80 | 33.47 | 644.84 | 19.37 | 4977 |
| limite | weibull | 100 | 3 | bossche | perfecto | 68.93 | 19.47 | 2552.06 | 11.99 | 4084 |
| limite | weibull | 100 | 4 | bossche | perfecto | 78.93 | 21.07 | 3422.25 | 4.40 | 4325 |
| limite | weibull | 100 | 5 | bossche | perfecto | 86.27 | 13.73 | 5685.51 | 3.92 | 3201 |
| limite | weibull | 100 | 6 | bossche | perfecto | 89.73 | 10.27 | 8018.33 | 3.70 | 2421 |
| limite | weibull | 100 | 7 | bossche | perfecto | 85.87 | 4.93 | 20108.66 | 12.61 | 1675 |
| limite | weibull | 100 | 8 | bossche | perfecto | 93.20 | 6.80 | 13542.52 | 3.47 | 1556 |
| limite | weibull | 100 | 9 | bossche | perfecto | 94.13 | 5.87 | 19278.88 | 3.37 | 1082 |
| limite | weibull | 100 | 2 | bossche | chebyshev | 42.27 | 8.93 | 35.42 | 15.47 | 2507 |
| limite | weibull | 100 | 3 | bossche | chebyshev | 78.40 | 21.60 | 218.63 | 5.10 | 13551 |
| limite | weibull | 100 | 4 | bossche | chebyshev | 90.00 | 10.00 | 940.50 | 4.23 | 10427 |
| limite | weibull | 100 | 5 | bossche | chebyshev | 90.00 | 10.00 | 2268.40 | 4.03 | 7168 |
| limite | weibull | 100 | 6 | bossche | chebyshev | 90.00 | 10.00 | 2902.61 | 3.91 | 5108 |
| limite | weibull | 100 | 7 | bossche | chebyshev | 90.13 | 9.87 | 3915.67 | 3.82 | 3819 |
| limite | weibull | 100 | 8 | bossche | chebyshev | 92.93 | 7.07 | 5102.69 | 3.64 | 2994 |
| limite | weibull | 100 | 9 | bossche | chebyshev | 93.87 | 6.13 | 6241.03 | 3.57 | 2483 |
| limite | weibull | 100 | 2 | pelaez | estimador05 | 42.40 | 53.47 | 39.31 | 0.30 | 271 |
| limite | weibull | 100 | 3 | pelaez | estimador05 | 89.47 | 4.80 | 263.63 | 0.41 | 296 |
| limite | weibull | 100 | 4 | pelaez | estimador05 | 93.87 | 2.67 | 251.12 | 0.89 | 334 |
| limite | weibull | 100 | 5 | pelaez | estimador05 | 91.47 | 3.73 | 902.49 | 0.75 | 339 |
| limite | weibull | 100 | 6 | pelaez | estimador05 | 93.20 | 2.27 | 2071.30 | 0.59 | 272 |
| limite | weibull | 100 | 7 | pelaez | estimador05 | 94.80 | 1.20 | 3903.12 | 0.25 | 235 |
| limite | weibull | 100 | 8 | pelaez | estimador05 | 94.80 | 0.80 | 5661.64 | 0.14 | 280 |
| limite | weibull | 100 | 9 | pelaez | estimador05 | 95.47 | 0.53 | 7637.45 | 0.04 | 245 |

Tabla B.2: Resultados análisis por límite temporal, carga Weibull, tiempo de aprovisionamiento de 100 segundos

| Análisis | Carga | Aprov. | Factor | Algortimo | Estimador | En plazo | Fuera plazo | Espera | Coste | VMs |
|---|---|---|---|---|---|---|---|---|---|---|
| limite | normal | 0 | 2 | pelaez | perfecto | 100.00 | 0.00 | 242.25 | 0.82 | 6373 |
| limite | normal | 0 | 2 | pelaez | media | 45.33 | 54.13 | 340.52 | 2.01 | 8989 |
| limite | normal | 0 | 2 | pelaez | chebyshev | 0.00 | 0.00 | 0.00 | 0.00 | 10 |
| limite | normal | 0 | 2 | pelaez | usuario | 51.33 | 40.93 | 358.47 | 1.90 | 10837 |
| limite | normal | 0 | 3 | pelaez | perfecto | 100.00 | 0.00 | 1112.15 | 0.67 | 3339 |
| limite | normal | 0 | 3 | pelaez | media | 65.73 | 33.47 | 1653.21 | 1.01 | 3464 |
| limite | normal | 0 | 3 | pelaez | chebyshev | 100.00 | 0.00 | 29.64 | 1.08 | 12772 |
| limite | normal | 0 | 3 | pelaez | usuario | 75.33 | 19.20 | 1245.83 | 1.07 | 6905 |
| limite | normal | 0 | 4 | pelaez | perfecto | 100.00 | 0.00 | 2568.95 | 0.59 | 1515 |
| limite | normal | 0 | 4 | pelaez | media | 77.07 | 22.27 | 3067.25 | 0.75 | 1629 |
| limite | normal | 0 | 4 | pelaez | chebyshev | 100.00 | 0.00 | 229.68 | 0.90 | 8597 |
| limite | normal | 0 | 4 | pelaez | usuario | 84.00 | 14.40 | 2658.65 | 0.76 | 2898 |
| limite | normal | 0 | 5 | pelaez | perfecto | 100.00 | 0.00 | 4206.58 | 0.56 | 705 |
| limite | normal | 0 | 5 | pelaez | media | 83.33 | 16.67 | 4632.52 | 0.66 | 821 |
| limite | normal | 0 | 5 | pelaez | chebyshev | 100.00 | 0.00 | 894.74 | 0.69 | 3886 |
| limite | normal | 0 | 5 | pelaez | usuario | 87.73 | 11.07 | 4037.09 | 0.67 | 1633 |
| limite | normal | 0 | 6 | pelaez | perfecto | 100.00 | 0.00 | 5593.32 | 0.55 | 460 |
| limite | normal | 0 | 6 | pelaez | media | 85.87 | 14.13 | 6220.82 | 0.63 | 502 |
| limite | normal | 0 | 6 | pelaez | chebyshev | 100.00 | 0.00 | 1649.12 | 0.64 | 2884 |
| limite | normal | 0 | 6 | pelaez | usuario | 90.93 | 8.53 | 5585.50 | 0.61 | 818 |
| limite | normal | 0 | 7 | pelaez | perfecto | 100.00 | 0.00 | 7087.73 | 0.54 | 296 |
| limite | normal | 0 | 7 | pelaez | media | 88.00 | 12.00 | 7400.02 | 0.58 | 351 |
| limite | normal | 0 | 7 | pelaez | chebyshev | 100.00 | 0.00 | 2683.16 | 0.60 | 1676 |
| limite | normal | 0 | 7 | pelaez | usuario | 88.53 | 10.40 | 6963.80 | 0.61 | 572 |
| limite | normal | 0 | 8 | pelaez | perfecto | 100.00 | 0.00 | 8416.19 | 0.53 | 225 |
| limite | normal | 0 | 8 | pelaez | media | 88.40 | 11.47 | 9584.65 | 0.61 | 423 |
| limite | normal | 0 | 8 | pelaez | chebyshev | 100.00 | 0.00 | 3497.97 | 0.57 | 905 |
| limite | normal | 0 | 8 | pelaez | usuario | 91.87 | 7.73 | 8998.77 | 0.59 | 461 |
| limite | normal | 0 | 9 | pelaez | perfecto | 100.00 | 0.00 | 10507.56 | 0.53 | 273 |
| limite | normal | 0 | 9 | pelaez | media | 90.67 | 9.33 | 10692.92 | 0.59 | 258 |
| limite | normal | 0 | 9 | pelaez | chebyshev | 100.00 | 0.00 | 4511.61 | 0.55 | 475 |
| limite | normal | 0 | 9 | pelaez | usuario | 94.27 | 5.07 | 9946.17 | 0.57 | 453 |

(*Continúa en la página siguiente*)

| Análisis | Carga | Aprov. | Factor | Algortimo | Estimador | En plazo | Fuera plazo | Espera | Coste | VMs |
|---|---|---|---|---|---|---|---|---|---|---|
| limite | normal | 0 | 2 | bossche | estimador05 | 20.80 | 79.20 | 143.56 | 1.33 | 414 |
| limite | normal | 0 | 3 | bossche | estimador05 | 83.47 | 16.40 | 884.93 | 0.40 | 622 |
| limite | normal | 0 | 4 | bossche | estimador05 | 88.93 | 11.07 | 1855.57 | 0.46 | 365 |
| limite | normal | 0 | 5 | bossche | estimador05 | 92.40 | 7.60 | 2723.93 | 0.49 | 273 |
| limite | normal | 0 | 6 | bossche | estimador05 | 93.20 | 6.80 | 4381.10 | 0.59 | 320 |
| limite | normal | 0 | 7 | bossche | estimador05 | 97.20 | 2.80 | 5768.26 | 0.54 | 218 |
| limite | normal | 0 | 8 | bossche | estimador05 | 95.87 | 4.13 | 7822.79 | 0.56 | 150 |
| limite | normal | 0 | 9 | bossche | estimador05 | 96.80 | 3.20 | 9426.03 | 0.54 | 156 |
| limite | normal | 0 | 2 | bossche | perfecto | 100.00 | 0.00 | 237.79 | 0.81 | 6129 |
| limite | normal | 0 | 3 | bossche | perfecto | 100.00 | 0.00 | 1114.63 | 0.67 | 3344 |
| limite | normal | 0 | 4 | bossche | perfecto | 100.00 | 0.00 | 2558.31 | 0.59 | 1550 |
| limite | normal | 0 | 5 | bossche | perfecto | 100.00 | 0.00 | 4174.66 | 0.55 | 657 |
| limite | normal | 0 | 6 | bossche | perfecto | 100.00 | 0.00 | 5703.72 | 0.55 | 520 |
| limite | normal | 0 | 7 | bossche | perfecto | 100.00 | 0.00 | 7005.42 | 0.54 | 266 |
| limite | normal | 0 | 8 | bossche | perfecto | 100.00 | 0.00 | 8653.19 | 0.53 | 213 |
| limite | normal | 0 | 9 | bossche | perfecto | 100.00 | 0.00 | 10549.90 | 0.53 | 273 |
| limite | normal | 0 | 2 | bossche | chebyshev | 0.00 | 0.00 | 0.00 | 0.00 | 10 |
| limite | normal | 0 | 3 | bossche | chebyshev | 100.00 | 0.00 | 29.70 | 1.08 | 12768 |
| limite | normal | 0 | 4 | bossche | chebyshev | 100.00 | 0.00 | 229.77 | 0.90 | 8596 |
| limite | normal | 0 | 5 | bossche | chebyshev | 100.00 | 0.00 | 892.40 | 0.69 | 3879 |
| limite | normal | 0 | 6 | bossche | chebyshev | 100.00 | 0.00 | 1646.26 | 0.64 | 2875 |
| limite | normal | 0 | 7 | bossche | chebyshev | 100.00 | 0.00 | 2687.81 | 0.59 | 1649 |
| limite | normal | 0 | 8 | bossche | chebyshev | 100.00 | 0.00 | 3507.39 | 0.57 | 896 |
| limite | normal | 0 | 9 | bossche | chebyshev | 100.00 | 0.00 | 4488.20 | 0.55 | 494 |
| limite | normal | 0 | 2 | pelaez | estimador05 | 22.80 | 77.20 | 154.75 | 1.45 | 480 |
| limite | normal | 0 | 3 | pelaez | estimador05 | 86.40 | 13.60 | 804.77 | 0.35 | 609 |
| limite | normal | 0 | 4 | pelaez | estimador05 | 92.13 | 7.87 | 1681.24 | 0.47 | 345 |
| limite | normal | 0 | 5 | pelaez | estimador05 | 94.67 | 5.33 | 2438.53 | 0.45 | 262 |
| limite | normal | 0 | 6 | pelaez | estimador05 | 93.73 | 6.27 | 3976.15 | 0.51 | 218 |
| limite | normal | 0 | 7 | pelaez | estimador05 | 96.27 | 3.73 | 5633.77 | 0.53 | 224 |
| limite | normal | 0 | 8 | pelaez | estimador05 | 96.13 | 3.87 | 7549.08 | 0.52 | 161 |
| limite | normal | 0 | 9 | pelaez | estimador05 | 96.27 | 3.73 | 9272.03 | 0.48 | 160 |

Tabla B.3: Resultados análisis por límite temporal, carga Normal, tiempo de aprovisionamiento de 0 segundos

| Análisis | Carga | Aprov. | Factor | Algortimo | Estimador | En plazo | Fuera plazo | Espera | Coste | VMs |
|---|---|---|---|---|---|---|---|---|---|---|
| limite | normal | 100 | 2 | pelaez | perfecto | 58.00 | 42.00 | 282.99 | 1.41 | 6358 |
| limite | normal | 100 | 2 | pelaez | media | 27.20 | 71.87 | 389.11 | 3.29 | 8444 |
| limite | normal | 100 | 2 | pelaez | chebyshev | 0.00 | 0.00 | 0.00 | 0.00 | 10 |
| limite | normal | 100 | 2 | pelaez | usuario | 28.80 | 63.47 | 420.65 | 3.38 | 10756 |
| limite | normal | 100 | 3 | pelaez | perfecto | 100.00 | 0.00 | 1143.47 | 0.67 | 3359 |
| limite | normal | 100 | 3 | pelaez | media | 63.33 | 35.47 | 1659.79 | 1.06 | 3569 |
| limite | normal | 100 | 3 | pelaez | chebyshev | 100.00 | 0.00 | 98.50 | 1.08 | 12621 |
| limite | normal | 100 | 3 | pelaez | usuario | 74.93 | 19.60 | 1289.92 | 1.07 | 6868 |
| limite | normal | 100 | 4 | pelaez | perfecto | 100.00 | 0.00 | 2579.28 | 0.59 | 1528 |
| limite | normal | 100 | 4 | pelaez | media | 76.67 | 22.67 | 3083.71 | 0.76 | 1650 |
| limite | normal | 100 | 4 | pelaez | chebyshev | 100.00 | 0.00 | 282.58 | 0.89 | 8433 |
| limite | normal | 100 | 4 | pelaez | usuario | 83.73 | 14.67 | 2679.48 | 0.76 | 2949 |
| limite | normal | 100 | 5 | pelaez | perfecto | 100.00 | 0.00 | 4210.37 | 0.56 | 701 |
| limite | normal | 100 | 5 | pelaez | media | 82.40 | 17.60 | 4667.99 | 0.67 | 814 |
| limite | normal | 100 | 5 | pelaez | chebyshev | 100.00 | 0.00 | 922.40 | 0.69 | 4010 |
| limite | normal | 100 | 5 | pelaez | usuario | 87.87 | 10.93 | 4070.90 | 0.67 | 1669 |
| limite | normal | 100 | 6 | pelaez | perfecto | 100.00 | 0.00 | 5605.48 | 0.55 | 481 |
| limite | normal | 100 | 6 | pelaez | media | 85.60 | 14.40 | 6162.11 | 0.64 | 510 |
| limite | normal | 100 | 6 | pelaez | chebyshev | 100.00 | 0.00 | 1677.99 | 0.64 | 2866 |
| limite | normal | 100 | 6 | pelaez | usuario | 90.80 | 8.67 | 5594.31 | 0.61 | 815 |
| limite | normal | 100 | 7 | pelaez | perfecto | 100.00 | 0.00 | 7101.40 | 0.54 | 338 |
| limite | normal | 100 | 7 | pelaez | media | 87.87 | 12.13 | 7663.49 | 0.62 | 462 |
| limite | normal | 100 | 7 | pelaez | chebyshev | 100.00 | 0.00 | 2705.31 | 0.60 | 1699 |
| limite | normal | 100 | 7 | pelaez | usuario | 88.53 | 10.40 | 6926.36 | 0.61 | 601 |
| limite | normal | 100 | 8 | pelaez | perfecto | 100.00 | 0.00 | 8799.51 | 0.54 | 296 |
| limite | normal | 100 | 8 | pelaez | media | 89.20 | 10.80 | 9220.61 | 0.60 | 303 |
| limite | normal | 100 | 8 | pelaez | chebyshev | 100.00 | 0.00 | 3551.40 | 0.57 | 889 |
| limite | normal | 100 | 8 | pelaez | usuario | 91.87 | 7.73 | 8960.67 | 0.59 | 475 |
| limite | normal | 100 | 9 | pelaez | perfecto | 100.00 | 0.00 | 10476.88 | 0.53 | 266 |
| limite | normal | 100 | 9 | pelaez | media | 90.53 | 9.47 | 10787.49 | 0.59 | 297 |
| limite | normal | 100 | 9 | pelaez | chebyshev | 100.00 | 0.00 | 4523.76 | 0.55 | 523 |
| limite | normal | 100 | 9 | pelaez | usuario | 94.00 | 5.33 | 9840.13 | 0.57 | 433 |

(*Continúa en la página siguiente*)

| Análisis | Carga | Aprov. | Factor | Algortimo | Estimador | En plazo | Fuera plazo | Espera | Coste | VMs |
|---|---|---|---|---|---|---|---|---|---|---|
| limite | normal | 100 | 2 | bossche | estimador05 | 20.80 | 79.20 | 206.57 | 2.10 | 593 |
| limite | normal | 100 | 3 | bossche | estimador05 | 80.80 | 19.20 | 913.93 | 0.40 | 629 |
| limite | normal | 100 | 4 | bossche | estimador05 | 88.93 | 11.07 | 2108.30 | 0.61 | 418 |
| limite | normal | 100 | 5 | bossche | estimador05 | 94.53 | 5.47 | 2457.51 | 0.47 | 280 |
| limite | normal | 100 | 6 | bossche | estimador05 | 96.93 | 3.07 | 3395.18 | 0.50 | 185 |
| limite | normal | 100 | 7 | bossche | estimador05 | 96.93 | 3.07 | 5477.41 | 0.53 | 195 |
| limite | normal | 100 | 8 | bossche | estimador05 | 96.40 | 3.60 | 7925.49 | 0.54 | 188 |
| limite | normal | 100 | 9 | bossche | estimador05 | 96.93 | 3.07 | 9474.22 | 0.55 | 151 |
| limite | normal | 100 | 2 | bossche | perfecto | 63.60 | 36.27 | 266.88 | 1.45 | 5556 |
| limite | normal | 100 | 3 | bossche | perfecto | 100.00 | 0.00 | 1135.18 | 0.67 | 3348 |
| limite | normal | 100 | 4 | bossche | perfecto | 100.00 | 0.00 | 2609.03 | 0.59 | 1561 |
| limite | normal | 100 | 5 | bossche | perfecto | 100.00 | 0.00 | 4144.80 | 0.55 | 638 |
| limite | normal | 100 | 6 | bossche | perfecto | 100.00 | 0.00 | 5689.61 | 0.55 | 514 |
| limite | normal | 100 | 7 | bossche | perfecto | 100.00 | 0.00 | 7002.78 | 0.54 | 297 |
| limite | normal | 100 | 8 | bossche | perfecto | 100.00 | 0.00 | 8625.23 | 0.53 | 215 |
| limite | normal | 100 | 9 | bossche | perfecto | 100.00 | 0.00 | 10583.11 | 0.53 | 273 |
| limite | normal | 100 | 2 | bossche | chebyshev | 0.00 | 0.00 | 0.00 | 0.00 | 10 |
| limite | normal | 100 | 3 | bossche | chebyshev | 100.00 | 0.00 | 98.53 | 1.08 | 12618 |
| limite | normal | 100 | 4 | bossche | chebyshev | 100.00 | 0.00 | 282.36 | 0.89 | 8432 |
| limite | normal | 100 | 5 | bossche | chebyshev | 100.00 | 0.00 | 918.00 | 0.69 | 4044 |
| limite | normal | 100 | 6 | bossche | chebyshev | 100.00 | 0.00 | 1682.05 | 0.64 | 2869 |
| limite | normal | 100 | 7 | bossche | chebyshev | 100.00 | 0.00 | 2717.49 | 0.59 | 1681 |
| limite | normal | 100 | 8 | bossche | chebyshev | 100.00 | 0.00 | 3528.43 | 0.57 | 886 |
| limite | normal | 100 | 9 | bossche | chebyshev | 100.00 | 0.00 | 4512.80 | 0.55 | 518 |
| limite | normal | 100 | 2 | pelaez | estimador05 | 18.93 | 81.07 | 243.98 | 1.86 | 629 |
| limite | normal | 100 | 3 | pelaez | estimador05 | 80.00 | 20.00 | 992.03 | 0.50 | 694 |
| limite | normal | 100 | 4 | pelaez | estimador05 | 90.27 | 9.73 | 1881.72 | 0.51 | 383 |
| limite | normal | 100 | 5 | pelaez | estimador05 | 94.00 | 6.00 | 2609.15 | 0.51 | 273 |
| limite | normal | 100 | 6 | pelaez | estimador05 | 94.13 | 5.87 | 4077.27 | 0.50 | 225 |
| limite | normal | 100 | 7 | pelaez | estimador05 | 96.27 | 3.73 | 5418.14 | 0.51 | 215 |
| limite | normal | 100 | 8 | pelaez | estimador05 | 96.27 | 3.73 | 7585.70 | 0.55 | 162 |
| limite | normal | 100 | 9 | pelaez | estimador05 | 97.20 | 2.80 | 9311.21 | 0.53 | 167 |

Tabla B.4: Resultados análisis por límite temporal, carga Normal, tiempo de aprovisionamiento de 100 segundos

| Análisis | Carga | Aprov. | Factor | Algortimo | Estimador | En plazo | Fuera plazo | Espera | Coste | VMs |
|---|---|---|---|---|---|---|---|---|---|---|
| limite | google | 0 | 2 | pelaez | perfecto | 100.00 | 0.00 | 18.15 | 4.39 | 143427 |
| limite | google | 0 | 2 | pelaez | media | 95.84 | 0.24 | 17.35 | 6.92 | 157405 |
| limite | google | 0 | 2 | pelaez | chebyshev | 90.15 | 0.49 | 15.49 | 13.27 | 136126 |
| limite | google | 0 | 2 | pelaez | usuario | 88.80 | 0.92 | 32.45 | 14.40 | 119020 |
| limite | google | 0 | 3 | pelaez | perfecto | 100.00 | 0.00 | 87.30 | 1.83 | 58460 |
| limite | google | 0 | 3 | pelaez | media | 97.98 | 0.00 | 80.72 | 5.50 | 88563 |
| limite | google | 0 | 3 | pelaez | chebyshev | 96.70 | 0.00 | 60.66 | 6.53 | 116226 |
| limite | google | 0 | 3 | pelaez | usuario | 94.43 | 0.55 | 108.89 | 8.08 | 62541 |
| limite | google | 0 | 4 | pelaez | perfecto | 100.00 | 0.00 | 152.39 | 1.04 | 32111 |
| limite | google | 0 | 4 | pelaez | media | 99.14 | 0.00 | 113.87 | 1.53 | 47867 |
| limite | google | 0 | 4 | pelaez | chebyshev | 99.02 | 0.00 | 94.43 | 2.64 | 73646 |
| limite | google | 0 | 4 | pelaez | usuario | 97.31 | 0.24 | 163.20 | 2.06 | 36992 |
| limite | google | 0 | 5 | pelaez | perfecto | 100.00 | 0.00 | 243.33 | 0.69 | 19306 |
| limite | google | 0 | 5 | pelaez | media | 99.27 | 0.00 | 196.46 | 0.98 | 28138 |
| limite | google | 0 | 5 | pelaez | chebyshev | 99.33 | 0.00 | 146.18 | 1.48 | 46440 |
| limite | google | 0 | 5 | pelaez | usuario | 98.41 | 0.00 | 240.07 | 1.03 | 22387 |
| limite | google | 0 | 6 | pelaez | perfecto | 100.00 | 0.00 | 318.96 | 0.59 | 15180 |
| limite | google | 0 | 6 | pelaez | media | 99.08 | 0.00 | 267.84 | 0.85 | 18770 |
| limite | google | 0 | 6 | pelaez | chebyshev | 99.39 | 0.00 | 197.04 | 1.00 | 30205 |
| limite | google | 0 | 6 | pelaez | usuario | 98.23 | 0.06 | 315.20 | 1.01 | 16857 |
| limite | google | 0 | 2 | bossche | perfecto | 100.00 | 0.00 | 14.11 | 4.43 | 144670 |
| limite | google | 0 | 3 | bossche | perfecto | 100.00 | 0.00 | 86.76 | 1.80 | 57623 |
| limite | google | 0 | 4 | bossche | perfecto | 100.00 | 0.00 | 151.97 | 1.03 | 32096 |
| limite | google | 0 | 5 | bossche | perfecto | 100.00 | 0.00 | 240.38 | 0.69 | 19277 |
| limite | google | 0 | 6 | bossche | perfecto | 100.00 | 0.00 | 319.35 | 0.59 | 15166 |
| limite | google | 0 | 2 | bossche | estimador05 | 88.19 | 11.69 | 89.87 | 2.93 | 79790 |
| limite | google | 0 | 3 | bossche | estimador05 | 97.92 | 1.96 | 143.03 | 1.24 | 37409 |
| limite | google | 0 | 4 | bossche | estimador05 | 99.39 | 0.61 | 218.92 | 0.86 | 23717 |
| limite | google | 0 | 5 | bossche | estimador05 | 99.69 | 0.24 | 295.81 | 0.68 | 16676 |
| limite | google | 0 | 6 | bossche | estimador05 | 99.94 | 0.06 | 380.83 | 0.61 | 14529 |
| limite | google | 0 | 2 | bossche | chebyshev | 90.76 | 0.37 | 9.27 | 11.34 | 121189 |
| limite | google | 0 | 3 | bossche | chebyshev | 96.76 | 0.00 | 58.50 | 6.51 | 115710 |

(*Continúa en la página siguiente*)

| Análisis | Carga | Aprov. | Factor | Algortimo | Estimador | En plazo | Fuera plazo | Espera | Coste | VMs |
|---|---|---|---|---|---|---|---|---|---|---|
| limite | google | 0 | 4 | bossche | chebyshev | 99.02 | 0.00 | 93.02 | 2.63 | 73628 |
| limite | google | 0 | 5 | bossche | chebyshev | 99.33 | 0.00 | 145.05 | 1.48 | 46251 |
| limite | google | 0 | 6 | bossche | chebyshev | 99.39 | 0.00 | 197.02 | 1.00 | 30164 |
| limite | google | 0 | 2 | pelaez | estimador05 | 86.35 | 12.85 | 95.24 | 2.83 | 74316 |
| limite | google | 0 | 3 | pelaez | estimador05 | 97.98 | 1.96 | 145.05 | 1.25 | 37548 |
| limite | google | 0 | 4 | pelaez | estimador05 | 99.39 | 0.49 | 220.17 | 0.86 | 23795 |
| limite | google | 0 | 5 | pelaez | estimador05 | 99.69 | 0.31 | 297.71 | 0.67 | 16567 |
| limite | google | 0 | 6 | pelaez | estimador05 | 99.82 | 0.18 | 382.82 | 0.61 | 14627 |

Tabla B.5: Resultados análisis por límite temporal, carga Google, tiempo de aprovisionamiento de 0 segundos

| Análisis | Carga | Aprov. | Ratio | Algortimo | Estimador | En plazo | Fuera plazo | Espera | Coste | VMs |
|---|---|---|---|---|---|---|---|---|---|---|
| llegada | weibull | 0 | 0002 | pelaez | perfecto | 100.00 | 0.00 | 3528.01 | 3.42 | 3142 |
| llegada | weibull | 0 | 0002 | pelaez | media | 89.20 | 10.40 | 3765.57 | 3.87 | 3871 |
| llegada | weibull | 0 | 0002 | pelaez | chebyshev | 100.00 | 0.00 | 881.19 | 3.81 | 10498 |
| llegada | weibull | 0 | 0002 | pelaez | usuario | 74.13 | 16.00 | 5186.99 | 4.72 | 5314 |
| llegada | weibull | 0 | 0200 | pelaez | perfecto | 100.00 | 0.00 | 11664.55 | 3.59 | 1688 |
| llegada | weibull | 0 | 0200 | pelaez | media | 90.67 | 9.20 | 7289.27 | 4.00 | 2388 |
| llegada | weibull | 0 | 0200 | pelaez | chebyshev | 100.00 | 0.00 | 2648.19 | 3.72 | 3767 |
| llegada | weibull | 0 | 0200 | pelaez | usuario | 92.40 | 5.73 | 9055.66 | 3.90 | 1960 |
| llegada | weibull | 0 | 2000 | pelaez | perfecto | 100.00 | 0.00 | 10100.68 | 3.60 | 1825 |
| llegada | weibull | 0 | 2000 | pelaez | media | 95.33 | 4.67 | 6317.43 | 3.82 | 2408 |
| llegada | weibull | 0 | 2000 | pelaez | chebyshev | 100.00 | 0.00 | 515.93 | 3.80 | 6551 |
| llegada | weibull | 0 | 2000 | pelaez | usuario | 97.07 | 2.13 | 8004.42 | 3.71 | 2147 |
| llegada | weibull | 0 | 4000 | pelaez | perfecto | 100.00 | 0.00 | 10103.13 | 3.60 | 1829 |
| llegada | weibull | 0 | 4000 | pelaez | media | 96.67 | 3.20 | 4435.12 | 3.79 | 2957 |
| llegada | weibull | 0 | 4000 | pelaez | chebyshev | 100.00 | 0.00 | 449.30 | 3.81 | 7102 |
| llegada | weibull | 0 | 4000 | pelaez | usuario | 98.00 | 1.60 | 8700.19 | 3.67 | 1998 |
| llegada | weibull | 0 | 6000 | pelaez | perfecto | 100.00 | 0.00 | 9871.91 | 3.60 | 1843 |
| llegada | weibull | 0 | 6000 | pelaez | media | 96.93 | 2.93 | 4451.61 | 3.78 | 2950 |
| llegada | weibull | 0 | 6000 | pelaez | chebyshev | 100.00 | 0.00 | 426.84 | 3.82 | 7322 |
| llegada | weibull | 0 | 6000 | pelaez | usuario | 98.53 | 1.33 | 8141.72 | 3.66 | 2068 |
| llegada | weibull | 0 | 8000 | pelaez | perfecto | 100.00 | 0.00 | 9648.08 | 3.60 | 1863 |
| llegada | weibull | 0 | 8000 | pelaez | media | 98.00 | 1.87 | 4480.06 | 3.73 | 2939 |
| llegada | weibull | 0 | 8000 | pelaez | chebyshev | 100.00 | 0.00 | 419.89 | 3.82 | 7325 |
| llegada | weibull | 0 | 8000 | pelaez | usuario | 98.53 | 1.33 | 7261.08 | 3.66 | 2236 |
| llegada | weibull | 0 | 0002 | bossche | estimador05 | 100.00 | 0.00 | 1364.04 | 3.31 | 2233 |
| llegada | weibull | 0 | 0200 | bossche | estimador05 | 49.87 | 50.13 | 19486.20 | 6.67 | 1015 |
| llegada | weibull | 0 | 2000 | bossche | estimador05 | 48.13 | 51.87 | 15185.31 | 7.40 | 1299 |
| llegada | weibull | 0 | 4000 | bossche | estimador05 | 53.33 | 46.67 | 16553.68 | 6.73 | 1234 |
| llegada | weibull | 0 | 6000 | bossche | estimador05 | 53.07 | 46.93 | 16707.00 | 6.82 | 1234 |
| llegada | weibull | 0 | 8000 | bossche | estimador05 | 53.07 | 46.93 | 16707.00 | 6.82 | 1234 |
| llegada | weibull | 0 | 0002 | bossche | perfecto | 100.00 | 0.00 | 3596.19 | 3.42 | 3150 |
| llegada | weibull | 0 | 0200 | bossche | perfecto | 100.00 | 0.00 | 11654.69 | 3.60 | 1688 |

(*Continúa en la página siguiente*)

| Análisis | Carga | Aprov. | Ratio | Algortimo | Estimador | En plazo | Fuera plazo | Espera | Coste | VMs |
|---|---|---|---|---|---|---|---|---|---|---|
| llegada | weibull | 0 | 2000 | bossche | perfecto | 100.00 | 0.00 | 10119.72 | 3.60 | 1823 |
| llegada | weibull | 0 | 4000 | bossche | perfecto | 100.00 | 0.00 | 10113.52 | 3.60 | 1827 |
| llegada | weibull | 0 | 6000 | bossche | perfecto | 100.00 | 0.00 | 9864.82 | 3.60 | 1844 |
| llegada | weibull | 0 | 8000 | bossche | perfecto | 100.00 | 0.00 | 9864.82 | 3.60 | 1844 |
| llegada | weibull | 0 | 0002 | bossche | chebyshev | 100.00 | 0.00 | 881.48 | 3.81 | 10458 |
| llegada | weibull | 0 | 0200 | bossche | chebyshev | 100.00 | 0.00 | 2659.78 | 3.72 | 3768 |
| llegada | weibull | 0 | 2000 | bossche | chebyshev | 100.00 | 0.00 | 484.01 | 3.81 | 6782 |
| llegada | weibull | 0 | 4000 | bossche | chebyshev | 100.00 | 0.00 | 449.30 | 3.81 | 7102 |
| llegada | weibull | 0 | 6000 | bossche | chebyshev | 100.00 | 0.00 | 426.84 | 3.82 | 7322 |
| llegada | weibull | 0 | 8000 | bossche | chebyshev | 100.00 | 0.00 | 419.89 | 3.82 | 7325 |
| llegada | weibull | 0 | 0002 | pelaez | estimador05 | 99.60 | 0.40 | 1345.80 | 3.38 | 2233 |
| llegada | weibull | 0 | 0200 | pelaez | estimador05 | 48.93 | 51.07 | 19683.01 | 6.93 | 1031 |
| llegada | weibull | 0 | 2000 | pelaez | estimador05 | 48.53 | 51.47 | 15319.88 | 7.20 | 1270 |
| llegada | weibull | 0 | 4000 | pelaez | estimador05 | 52.80 | 47.20 | 16615.26 | 6.73 | 1216 |
| llegada | weibull | 0 | 6000 | pelaez | estimador05 | 53.07 | 46.93 | 16631.31 | 6.79 | 1230 |
| llegada | weibull | 0 | 8000 | pelaez | estimador05 | 54.93 | 45.07 | 16693.46 | 6.55 | 1228 |

Tabla B.6: Resultados análisis por ratio de llegada, carga Weibull, tiempo de aprovisionamiento de 0 segundos

| Análisis | Carga | Aprov. | Ratio | Algortimo | Estimador | En plazo | Fuera plazo | Espera | Coste | VMs |
|---|---|---|---|---|---|---|---|---|---|---|
| llegada | weibull | 100 | 0002 | pelaez | perfecto | 79.07 | 20.93 | 3327.76 | 4.41 | 4328 |
| llegada | weibull | 100 | 0002 | pelaez | media | 66.53 | 33.33 | 3668.20 | 5.21 | 5394 |
| llegada | weibull | 100 | 0002 | pelaez | chebyshev | 90.00 | 10.00 | 942.77 | 4.23 | 10427 |
| llegada | weibull | 100 | 0002 | pelaez | usuario | 64.00 | 26.13 | 5185.99 | 5.54 | 6090 |
| llegada | weibull | 100 | 0200 | pelaez | perfecto | 95.60 | 4.13 | 11486.44 | 3.87 | 1738 |
| llegada | weibull | 100 | 0200 | pelaez | media | 86.80 | 12.67 | 6425.82 | 4.19 | 2633 |
| llegada | weibull | 100 | 0200 | pelaez | chebyshev | 91.87 | 8.13 | 2592.99 | 4.05 | 3771 |
| llegada | weibull | 100 | 0200 | pelaez | usuario | 86.80 | 11.47 | 9585.50 | 4.14 | 1899 |
| llegada | weibull | 100 | 2000 | pelaez | perfecto | 85.87 | 14.00 | 10046.86 | 4.25 | 1850 |
| llegada | weibull | 100 | 2000 | pelaez | media | 92.93 | 7.07 | 6230.19 | 3.92 | 2440 |
| llegada | weibull | 100 | 2000 | pelaez | chebyshev | 93.73 | 6.27 | 834.16 | 4.03 | 5478 |
| llegada | weibull | 100 | 2000 | pelaez | usuario | 92.80 | 6.40 | 8654.94 | 3.88 | 2013 |
| llegada | weibull | 100 | 4000 | pelaez | perfecto | 77.47 | 22.53 | 7741.96 | 4.66 | 2146 |
| llegada | weibull | 100 | 4000 | pelaez | media | 87.47 | 12.53 | 4752.57 | 4.18 | 2875 |
| llegada | weibull | 100 | 4000 | pelaez | chebyshev | 90.80 | 9.20 | 883.22 | 4.16 | 5472 |
| llegada | weibull | 100 | 4000 | pelaez | usuario | 90.00 | 9.60 | 5994.61 | 4.02 | 2543 |
| llegada | weibull | 100 | 6000 | pelaez | perfecto | 79.33 | 20.67 | 6326.97 | 4.57 | 2442 |
| llegada | weibull | 100 | 6000 | pelaez | media | 90.40 | 9.47 | 5208.32 | 4.04 | 2707 |
| llegada | weibull | 100 | 6000 | pelaez | chebyshev | 86.53 | 13.47 | 1709.63 | 4.32 | 4413 |
| llegada | weibull | 100 | 6000 | pelaez | usuario | 87.87 | 12.00 | 5562.51 | 4.12 | 2676 |
| llegada | weibull | 100 | 8000 | pelaez | perfecto | 79.07 | 20.13 | 6208.67 | 5.00 | 2590 |
| llegada | weibull | 100 | 8000 | pelaez | media | 89.07 | 10.80 | 3799.52 | 4.12 | 3239 |
| llegada | weibull | 100 | 8000 | pelaez | chebyshev | 85.60 | 14.40 | 1035.47 | 4.40 | 5121 |
| llegada | weibull | 100 | 8000 | pelaez | usuario | 86.67 | 13.20 | 5519.65 | 4.18 | 2690 |
| llegada | weibull | 100 | 0002 | bossche | estimador05 | 92.67 | 2.40 | 356.65 | 5.26 | 299 |
| llegada | weibull | 100 | 0200 | bossche | estimador05 | 51.07 | 47.20 | 20193.65 | 6.45 | 876 |
| llegada | weibull | 100 | 2000 | bossche | estimador05 | 42.93 | 45.07 | 17259.43 | 9.84 | 766 |
| llegada | weibull | 100 | 4000 | bossche | estimador05 | 65.33 | 19.33 | 12594.33 | 7.06 | 1140 |
| llegada | weibull | 100 | 6000 | bossche | estimador05 | 68.80 | 16.13 | 10978.81 | 6.48 | 1304 |
| llegada | weibull | 100 | 8000 | bossche | estimador05 | 68.80 | 16.13 | 10978.81 | 6.48 | 1304 |
| llegada | weibull | 100 | 0002 | bossche | perfecto | 78.93 | 21.07 | 3296.16 | 4.41 | 4317 |
| llegada | weibull | 100 | 0200 | bossche | perfecto | 96.80 | 2.80 | 11471.87 | 3.85 | 1750 |

(*Continúa en la página siguiente*)

| Análisis | Carga | Aprov. | Ratio | Algortimo | Estimador | En plazo | Fuera plazo | Espera | Coste | VMs |
|---|---|---|---|---|---|---|---|---|---|---|
| llegada | weibull | 100 | 2000 | bossche | perfecto | 85.87 | 14.00 | 10065.64 | 4.25 | 1848 |
| llegada | weibull | 100 | 4000 | bossche | perfecto | 77.60 | 22.40 | 7725.68 | 4.66 | 2148 |
| llegada | weibull | 100 | 6000 | bossche | perfecto | 79.07 | 20.93 | 6324.35 | 4.58 | 2442 |
| llegada | weibull | 100 | 8000 | bossche | perfecto | 79.07 | 20.93 | 6324.35 | 4.58 | 2442 |
| llegada | weibull | 100 | 0002 | bossche | chebyshev | 90.00 | 10.00 | 940.71 | 4.23 | 10427 |
| llegada | weibull | 100 | 0200 | bossche | chebyshev | 91.60 | 8.40 | 2579.21 | 4.06 | 3771 |
| llegada | weibull | 100 | 2000 | bossche | chebyshev | 93.87 | 6.13 | 837.87 | 4.03 | 5479 |
| llegada | weibull | 100 | 4000 | bossche | chebyshev | 90.80 | 9.20 | 883.22 | 4.16 | 5472 |
| llegada | weibull | 100 | 6000 | bossche | chebyshev | 86.53 | 13.47 | 1709.63 | 4.32 | 4413 |
| llegada | weibull | 100 | 8000 | bossche | chebyshev | 85.60 | 14.40 | 1035.47 | 4.40 | 5121 |
| llegada | weibull | 100 | 0002 | pelaez | estimador05 | 90.93 | 4.67 | 399.88 | 0.53 | 383 |
| llegada | weibull | 100 | 0200 | pelaez | estimador05 | 48.27 | 50.53 | 20093.99 | 6.50 | 952 |
| llegada | weibull | 100 | 2000 | pelaez | estimador05 | 44.27 | 44.27 | 17378.52 | 4.89 | 758 |
| llegada | weibull | 100 | 4000 | pelaez | estimador05 | 64.27 | 19.33 | 12956.82 | 3.97 | 1141 |
| llegada | weibull | 100 | 6000 | pelaez | estimador05 | 68.93 | 13.07 | 11133.95 | 3.57 | 1289 |
| llegada | weibull | 100 | 8000 | pelaez | estimador05 | 69.47 | 14.13 | 10709.57 | 3.21 | 1341 |

Tabla B.7: Resultados análisis por ratio de llegada, carga Weibull, tiempo de aprovisionamiento de 100 segundos

| Análisis | Carga | Aprov. | Ratio | Algortimo | Estimador | En plazo | Fuera plazo | Espera | Coste | VMs |
|---|---|---|---|---|---|---|---|---|---|---|
| llegada | normal | 100 | 0002 | pelaez | perfecto | 100.00 | 0.00 | 2579.28 | 0.59 | 1528 |
| llegada | normal | 100 | 0002 | pelaez | media | 76.67 | 22.67 | 3083.71 | 0.76 | 1650 |
| llegada | normal | 100 | 0002 | pelaez | chebyshev | 100.00 | 0.00 | 282.58 | 0.89 | 8433 |
| llegada | normal | 100 | 0002 | pelaez | usuario | 83.73 | 14.67 | 2679.48 | 0.76 | 2949 |
| llegada | normal | 100 | 0200 | pelaez | perfecto | 100.00 | 0.00 | 3419.52 | 0.91 | 3885 |
| llegada | normal | 100 | 0200 | pelaez | media | 86.00 | 14.00 | 3312.00 | 1.06 | 3889 |
| llegada | normal | 100 | 0200 | pelaez | chebyshev | 100.00 | 0.00 | 960.17 | 1.03 | 7242 |
| llegada | normal | 100 | 0200 | pelaez | usuario | 97.33 | 2.53 | 2909.79 | 0.93 | 3730 |
| llegada | normal | 100 | 2000 | pelaez | perfecto | 100.00 | 0.00 | 3134.16 | 0.92 | 4230 |
| llegada | normal | 100 | 2000 | pelaez | media | 98.67 | 1.20 | 1709.82 | 0.97 | 5670 |
| llegada | normal | 100 | 2000 | pelaez | chebyshev | 100.00 | 0.00 | 430.93 | 1.11 | 11591 |
| llegada | normal | 100 | 2000 | pelaez | usuario | 99.33 | 0.67 | 2516.89 | 0.94 | 4137 |
| llegada | normal | 100 | 4000 | pelaez | perfecto | 100.00 | 0.00 | 3121.65 | 0.92 | 4220 |
| llegada | normal | 100 | 4000 | pelaez | media | 98.27 | 1.73 | 1594.64 | 0.99 | 5849 |
| llegada | normal | 100 | 4000 | pelaez | chebyshev | 100.00 | 0.00 | 642.12 | 1.04 | 9357 |
| llegada | normal | 100 | 4000 | pelaez | usuario | 99.60 | 0.40 | 2419.77 | 0.95 | 4234 |
| llegada | normal | 100 | 6000 | pelaez | perfecto | 100.00 | 0.00 | 3105.80 | 0.92 | 4228 |
| llegada | normal | 100 | 6000 | pelaez | media | 98.40 | 1.60 | 1512.56 | 1.00 | 5927 |
| llegada | normal | 100 | 6000 | pelaez | chebyshev | 100.00 | 0.00 | 435.75 | 1.10 | 11315 |
| llegada | normal | 100 | 6000 | pelaez | usuario | 99.73 | 0.27 | 2376.34 | 0.95 | 4293 |
| llegada | normal | 100 | 8000 | pelaez | perfecto | 100.00 | 0.00 | 3101.86 | 0.92 | 4232 |
| llegada | normal | 100 | 8000 | pelaez | media | 98.67 | 1.33 | 1508.83 | 1.00 | 5919 |
| llegada | normal | 100 | 8000 | pelaez | chebyshev | 100.00 | 0.00 | 410.77 | 1.11 | 11594 |
| llegada | normal | 100 | 8000 | pelaez | usuario | 98.93 | 1.07 | 2350.44 | 0.96 | 4331 |
| llegada | normal | 100 | 0002 | bossche | estimador05 | 89.20 | 10.80 | 2043.49 | 0.55 | 399 |
| llegada | normal | 100 | 0200 | bossche | estimador05 | 9.33 | 90.53 | 5544.14 | 8.96 | 2382 |
| llegada | normal | 100 | 2000 | bossche | estimador05 | 22.80 | 77.07 | 4806.80 | 3.69 | 2680 |
| llegada | normal | 100 | 4000 | bossche | estimador05 | 26.80 | 73.20 | 4755.72 | 3.19 | 2730 |
| llegada | normal | 100 | 6000 | bossche | estimador05 | 30.00 | 69.87 | 4733.65 | 2.86 | 2760 |
| llegada | normal | 100 | 8000 | bossche | estimador05 | 30.00 | 69.87 | 4733.65 | 2.86 | 2760 |
| llegada | normal | 100 | 0002 | bossche | perfecto | 100.00 | 0.00 | 2565.08 | 0.59 | 1562 |
| llegada | normal | 100 | 0200 | bossche | perfecto | 100.00 | 0.00 | 3421.34 | 0.91 | 3881 |

(*Continúa en la página siguiente*)

| Análisis | Carga | Aprov. | Ratio | Algortimo | Estimador | En plazo | Fuera plazo | Espera | Coste | VMs |
|---|---|---|---|---|---|---|---|---|---|---|
| llegada | normal | 100 | 2000 | bossche | perfecto | 100.00 | 0.00 | 3133.21 | 0.92 | 4230 |
| llegada | normal | 100 | 4000 | bossche | perfecto | 100.00 | 0.00 | 3117.91 | 0.92 | 4226 |
| llegada | normal | 100 | 6000 | bossche | perfecto | 100.00 | 0.00 | 3105.73 | 0.92 | 4228 |
| llegada | normal | 100 | 8000 | bossche | perfecto | 100.00 | 0.00 | 3105.73 | 0.92 | 4228 |
| llegada | normal | 100 | 0002 | bossche | chebyshev | 100.00 | 0.00 | 282.36 | 0.89 | 8432 |
| llegada | normal | 100 | 0200 | bossche | chebyshev | 100.00 | 0.00 | 1009.94 | 1.02 | 7264 |
| llegada | normal | 100 | 2000 | bossche | chebyshev | 100.00 | 0.00 | 431.08 | 1.11 | 11590 |
| llegada | normal | 100 | 4000 | bossche | chebyshev | 100.00 | 0.00 | 642.12 | 1.04 | 9357 |
| llegada | normal | 100 | 6000 | bossche | chebyshev | 100.00 | 0.00 | 435.75 | 1.10 | 11315 |
| llegada | normal | 100 | 8000 | bossche | chebyshev | 100.00 | 0.00 | 410.77 | 1.11 | 11594 |
| llegada | normal | 100 | 0002 | pelaez | estimador05 | 91.07 | 8.93 | 1909.46 | 0.47 | 383 |
| llegada | normal | 100 | 0200 | pelaez | estimador05 | 9.07 | 90.93 | 5529.71 | 9.27 | 2362 |
| llegada | normal | 100 | 2000 | pelaez | estimador05 | 21.47 | 78.53 | 4795.77 | 3.96 | 2701 |
| llegada | normal | 100 | 4000 | pelaez | estimador05 | 24.80 | 75.20 | 4750.74 | 3.38 | 2685 |
| llegada | normal | 100 | 6000 | pelaez | estimador05 | 28.13 | 71.87 | 4747.81 | 3.04 | 2708 |
| llegada | normal | 100 | 8000 | pelaez | estimador05 | 26.13 | 73.87 | 4735.89 | 3.25 | 2717 |

Tabla B.8: Resultados análisis por ratio de llegada, carga Normal, tiempo de aprovisionamiento de 100 segundos

# Bibliografía


[1] P. Mell and T. Grance, "The NIST definition of cloud computing (draft)," *NIST special publication*, 2011. 1, 7

[2] M. Mahjoub, A. Mdhaffar, R. B. Halima, and M. Jmaiel, "A Comparative Study of the Current Cloud Computing Technologies and Offers," *2011 First International Symposium on Network Cloud Computing and Applications*, pp. 131–134, nov 2011. 1

[3] Q. Zhang, L. Cheng, and R. Boutaba, "Cloud computing: State-of-the-art and research challenges," *Journal of Internet Services and Applications*, vol. 1, no. 1, pp. 7–18, 2010. 1

[4] S. Shenai, "Survey on Scheduling Issues in Cloud Computing," *Procedia Engineering*, vol. 38, pp. 2881–2888, jan 2012. 2, 3, 10

[5] T. N. Minh, T. Nam, and D. H. J. Epema, "Parallel workload modeling with realistic characteristics," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 8, pp. 2138–2148, 2014. 2

[6] F. A. Da Silva and H. Senger, "Scalability limits of Bag-of-Tasks applications running on hierarchical platforms," *Journal of Parallel and Distributed Computing*, vol. 71, no. 6, pp. 788–801, 2011. 3

[7] A. Iosup and D. Epema, "Grid Computing Workloads : Bags of Tasks , Workflows , Pilots , and Others," *IEEE Internet Computing*, vol. 15, no. 2, pp. 19–26, 2011. 3

[8] S. Singh and I. Chana, "A Survey on Resource Scheduling in Cloud Computing: Issues and Challenges," *Journal of Grid Computing*, 2016. 3

[9] L. Thai, B. Varghese, and A. Barker, "A survey and taxonomy of resource optimisation for executing bag-of-task applications on public clouds," *Future Generation Computer Systems*, vol. 82, no. May, pp. 1–11, 2018. 3

[10] C. B. Lee and a. Snavely, "On the User-Scheduler Dialogue: Studies of User-Provided Runtime Estimates and Utility Functions," *International Journal of High Performance Computing Applications*, vol. 20, no. 4, pp. 495–506, 2006. 4, 21

[11] R. N. Calheiros and R. Buyya, "Meeting deadlines of scientific workflows in public clouds with tasks replication," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 7, pp. 1787–1796, 2014. 4, 19, 64

[12] org Schad, J. Dittrich, J.-A. Quia e Ruiz, J. Schad, J. Dittrich, and J.-A. Quiané-Ruiz, "Runtime measurements in the cloud: observing, analyzing, and reducing variance," *Proc. VLDB Endow.*, vol. 3, no. 1-2, pp. 460–471, 2010. 4

[13] A. Iosup, N. Yigitbasi, and D. Epema, "On the performance variability of production cloud services," *Proceedings - 11th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing, CCGrid 2011*, no. January, pp. 104–113, 2011. 4

[14] M. L. Pinedo, "Parallel Machine Models (Deterministic)," in *Scheduling: Theory, Algorithms, and Systems*, pp. 111–149, Springer-Verlag New York, 4 ed., 2012. 10

[15] R. Van Den Bossche, K. Vanmechelen, and J. Broeckhove, "Online cost-efficient scheduling of deadline-constrained workloads on hybrid clouds," *Future Generation Computer Systems*, vol. 29, no. 4, pp. 973–985, 2013. 10, 18, 19, 20, 37, 60, 61, 64, 66, 67, 116

[16] B. Lin, W. Guo, and X. Lin, "Online optimization scheduling for scientific workflows with deadline constraint on hybrid clouds," *Concurrency Computation Practice and Experience*, vol. 28, no. 6, pp. 3079–3095, 2016. 11, 18, 20

[17] W.-J. Wang, Y.-S. Chang, W.-T. Lo, and Y.-K. Lee, "Adaptive scheduling for parallel tasks with QoS satisfaction for hybrid cloud environments," *The Journal of Supercomputing*, feb 2013. 12, 18, 20

[18] J. Celaya and U. Arronategui, "Fair scheduling of bag-of-tasks applications on large-scale platforms," *Future Generation Computer Systems*, vol. 49, pp. 28–44, 2015. 13, 18, 20

[19] I. a. Moschakis and H. D. Karatza, "A meta-heuristic optimization approach to the scheduling of bag-of-tasks applications on heterogeneous clouds with multi-level arrivals and critical jobs," *Simulation Modelling Practice and Theory*, vol. 57, pp. 1–25, 2015. 13, 18, 20

[20] Z. Cai, X. Li, R. Ruiz, and Q. Li, "A delay-based dynamic scheduling algorithm for bag-of-task workflows with stochastic task execution times in clouds," *Future Generation Computer Systems*, vol. 71, pp. 57–72, 2017. 13, 18, 20

[21] Z. Cai, X. Li, and R. Ruiz, "Cloud yarn resource provisioning for task-batch based workflows with deadlines," tech. rep., Technical report https://github.com/czcnjust/elasticsim/blob/master/technicalreport201500805. pdf, 2016. 13

[22] J. J. Durillo and R. Prodan, "Multi-objective workflow scheduling in amazon ec2," *Cluster Computing*, vol. 17, pp. 169–189, June 2014. 14

[23] A.-M. Oprescu, T. Kielmann, and H. Leahu, "Budget Estimation and Control for Bag-of-Tasks Scheduling in Clouds," *Parallel Processing Letters*, vol. 21, no. 02, pp. 219–243, 2011. 14, 18, 20

[24] H. Arabnejad, J. G. Barbosa, and R. Prodan, "Low-time complexity budget – deadline constrained workflow scheduling on heterogeneous resources," *Future Generation Computer Systems*, vol. 55, pp. 29–40, 2016. 14, 18, 20

[25] R. Prodan and M. Wieczorek, "Bi-criteria scheduling of scientific grid workflows," *IEEE Transactions on Automation Science and Engineering*, vol. 7, no. 2, pp. 364–376, 2010. 15

[26] R. Sakellariou, H. Zhao, E. Tsiakkouri, and M. D. Dikaiakos, "Scheduling workflows with budget constraints," in *Integrated research in GRID computing*, pp. 189–202, Springer, 2007. 15

[27] J. Yu and R. Buyya, "Scheduling scientific workflow applications with deadline and budget constraints using genetic algorithms," *Scientific Programming*, vol. 14, no. 3-4, pp. 217–230, 2006. 15

[28] W. Zheng and R. Sakellariou, "Budget-deadline constrained workflow planning for admission control in market-oriented environments," in *International Workshop on Grid Economics and Business Models*, pp. 105–119, Springer, 2011. 15

[29] I. a. Moschakis and H. D. Karatza, "Evaluation of gang scheduling performance and cost in a cloud computing system," *The Journal of Supercomputing*, vol. 59, pp. 975–992, sep 2010. 15, 18, 20

[30] I. a. Moschakis and H. D. Karatza, "Multi-criteria scheduling of Bag-of-Tasks applications on heterogeneous interlinked clouds with simulated annealing," *Journal of Systems and Software*, vol. 101, pp. 1–14, 2015. 15

[31] L. Wu, S. Kumar Garg, and R. Buyya, "SLA-based admission control for a Software-as-a-Service provider in Cloud computing environments," *Journal of Computer and System Sciences*, vol. 78, pp. 1280–1299, sep 2012. 15, 18, 20, 68

[32] R. Calheiros and R. Buyya, "Cost-Effective provisioning and scheduling of deadline-constrained applications in hybrid clouds," *Web Information Systems Engineering-WISE ...*, 2012. 16, 18, 20

[33] R. L. R. L. Graham, "Bounds on Multiprocessing Timing Anomalies R.," *SIAM Journal on Computing*, vol. 17, no. 2, pp. 416–429, 1969. 16

[34] J. M. J. Schutten, "List scheduling revisited," *Operations Research Letters*, vol. 18, no. 4, pp. 167–170, 1996. 16

[35] H. Arabnejad and J. G. Barbosa, "List scheduling algorithm for heterogeneous systems by an optimistic cost table," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 3, pp. 682–694, 2014. 17

[36] M. Malawski, G. Juve, E. Deelman, and J. Nabrzyski, "Cost- and deadline-constrained provisioning for scientific workflow ensembles in IaaS clouds," *International Conference for High Performance Computing, Networking, Storage and Analysis, SC*, vol. 48, pp. 1–18, 2012. 19

[37] J. Shi, J. Luo, F. Dong, J. Zhang, and J. Zhang, "Elastic resource provisioning for scientific workflow scheduling in cloud under budget and deadline constraints," *Cluster Computing*, 2016. 21

[38] A. AuYoung, A. Vahdat, and A. C. Snoeren, "Evaluating the impact of inaccurate information in utility-based scheduling," *Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis - SC '09*, p. 1, 2009. 21

[39] D. Tsafrir, Y. Etsion, and D. G. Feitelson, "Backfilling Using System-Generated Predictions Rather than User Runtime Estimates," *IEEE Transactions on Parallel and Distributed Systems*, vol. 18, pp. 789–803, jun 2007. 21, 50

[40] K. R. Jackson, L. Ramakrishnan, K. Muriki, S. Canon, S. Cholia, J. Shalf, H. J. Wasserman, and N. J. Wright, "Performance Analysis of High Performance Computing Applications on the Amazon Web Services Cloud," *2nd IEEE International Conference on Cloud Computing Technology and Science*, pp. 159–168, 2010. 21

[41] S. Verboven, P. Hellinckx, F. Arickx, and J. Broeckhove, "Runtime prediction based grid scheduling of parameter sweep jobs," *Journal of Internet Technology*, vol. 11, no. 1, pp. 47–54, 2010. 22

[42] R. D. Silva, G. Juve, and E. Deelman, "Toward fine-grained online task characteristics estimation in scientific workflows," *... Workflows in Support of ...*, 2013. 22

[43] I. Pietri, G. Juve, E. Deelman, and R. Sakellariou, "A Performance Model to Estimate Execution Time of Scientific Workflows on the Cloud," in *Workflows in Support of Large-Scale Science (WORKS), 2014 9th Workshop on*, 2014. 22

[44] R. L. Graham, E. L. Lawler, J. K. Lenstra, and a. H. G. R. Kan, "Optimization and Approximation in Deterministic Sequencing and Scheduling: a Survey," 1979. 30

[45] M. L. Pinedo, *Scheduling*. 2012. 32, 34

[46] C. Koulamas, "The Total Tardiness Problem: Review and Extensions," *Operations Research*, vol. 42, no. 6, pp. 1025–1041, 1994. 33, 34

[47] B. Alidaee and D. Rosa, "Scheduling parallel machines to minimize total weighted and unweighted tardiness," *Computers & Operations Research*, vol. 24, no. 8, pp. 775–788, 1997. 33

[48] J. K. Lenstra, a. H. G. Rinnooy Kan, and P. Brucker, "Complexity of Machine Scheduling Problems," 1977. 33

[49] J. A. Stankovic, M. Spuri, M. D. Natale, and S. S. S. Anna, "Implications of Classical Scheduling Results For Real-Time Systems," *Science*, pp. 1–24, 1994. 34, 37

[50] M. L. Dertouzos and A. K. Mok, "Multiprocessor on-line scheduling of hard-real-time tasks," *IEEE Transactions on Software Engineering*, vol. 15, no. 12, pp. 1497–1506, 1989. 37

[51] M. Mao and M. Humphrey, "A Performance Study on the VM Startup Time in the Cloud," *2012 IEEE Fifth International Conference on Cloud Computing*, pp. 423–430, jun 2012. 42

[52] P. Gaenssler and J. A. Wellner, "Glivenko-cantelli theorems," *Wiley StatsRef: Statistics Reference Online*, 2014. 49

[53] K. Pearson, "Method of moments and method of maximum likelihood," *Biometrika*, vol. 28, no. 1/2, pp. 34–59, 1936. 49

[54] D. G. Feitelson and B. Nitzberg, "Job Characteristics of a Production Parallel Scientific Workload on the {NASA Ames iPSC/860}," *Job Scheduling Strategies for Parallel Processing*, pp. 337–360, 1995. 49

[55] R. Gibbons, "A Historical Application Profiler for Use by Parallel Schedulers," *Job Scheduling Strategies for Parallel Processing Lecture Notes in Computer Science*, vol. Volume 129, pp. pp 58–77, 1997. 49

[56] A. Kabán, "Non-parametric detection of meaningless distances in high dimensional data," *Statistics and Computing*, vol. 22, no. 2, pp. 375–385, 2012. 55

[57] M. Hazewinkel, "Chebyshev inequality in probability theory," in *Encyclopedia of Mathematics*, Springer, 2001. 55

[58] R. Wilhelm, J. Engblom, A. Ermedahl, N. Holsti, S. Thesing, D. Whalley, G. Bernat, C. Ferdinand, R. Heckmann, T. Mitra, F. Mueller, I. Puaut, P. Pushner, J. Staschular, and P. Stenstrom, "The Worst-Case Execution Time Problem — Overview of Methods and Survey of Tools," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 7, no. 3, pp. 1–52, 2008. 57

[59] R. Calheiros and R. Ranjan, "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Software: Practice and Experience*, no. August 2010, pp. 23–50, 2011. 61, 68

[60] K. Sarda, S. Sanghrajka, and R. Sion, "Cloud Performance Benchmark Series," 2011. 64

[61] C. Reiss, J. Wilkes, and J. L. Hellerstein, "Google cluster-usage traces: format + schema," 2012. 66

[62] A. Iosup, O. Sonmez, S. Anoep, and D. Epema, "The performance of bags-of-tasks in large-scale distributed systems," *Proceedings of the 17th international symposium on High performance distributed computing - HPDC '08*, no. May 2014, p. 97, 2008. 67

[63] A. M. Oprescu and T. Kielmann, "Bag-of-tasks scheduling under budget constraints," *Proceedings - 2nd IEEE International Conference on Cloud Computing Technology and Science, CloudCom 2010*, pp. 351–359, 2010. 68

[64] C. Reiss, J. Wilkes, and J. L. Hellerstein, "Obfuscatory obscanturism: Making workload traces of commercially-sensitive systems safe to release," *Proceedings of the 2012 IEEE Network Operations and Management Symposium, NOMS 2012*, pp. 1279–1286, 2012. 69

[65] S. Di, D. Kondo, and F. Cappello, "Characterizing Cloud Applications on a Google Data Center," *2013 42nd International Conference on Parallel Processing*, pp. 468–473, oct 2013. 69

[66] C. Reiss, U. C. Berkeley, A. T. Cmu, G. R. G. Cmu, R. H. Katz, M. A. Kozuch, and I. Labs, "Towards understanding heterogeneous clouds at scale : Google trace analysis," 2012. 69

[67] S. E. Maxwell, N. Dame, and H. D. Delaney, *DESIGNING EXPERIMENTS AND ANALYZING DATA: A Model Comparison Perspective*. London: Lawrence Erlabaum Associates, Publishers, second edi ed., 2004. 73

[68] M. J. Crawley, "The R Book. Model Formulae in R," in *The R Book*, ch. Statistica, pp. 329–336, London: John Wiley & Sons, Ltd, 2007. 73

[69] J. M. Chambers and T. J. Hastie, "Statistical models. Chapter 2 of Statistical Models in S," in *Statistical models*, Wadsworth & Brooks/Cole, 1992. 73

[70] M. J. Blanca, R. Alarcón, J. Arnau, R. Bono, and R. Bendayan, "Non-normal data: Is ANOVA still a valid option?," *Psicotherma*, vol. 29, no. 4, pp. 552–557, 2017. 75

[71] J. W. Tukey, "Comparing individual means in the analysis of variance," *Biometrics*, vol. 5, no. 2, pp. 99–114, 1949. 75